Osborne McGraw-Hill

Covers All Versions, Including DOS 6

DOS

the COMPLETE REFERENCE

FOURTH EDITION

Best-Seller! Over 350,000 Copies Sold

KRIS JAMSA

The Most Thorough Discussion of MS-DOS and PC-DOS Available – A Must for Any PC User

DOS: The Complete Reference,
Fourth Edition

About the Author...

Kris Jamsa is the author of more than 40 computer books, and has published on a wide variety of computer topics, with expertise in DOS, Windows, hard disk management, graphics, WordPerfect, and programming languages.

Kris graduated from the United States Air Force Academy in 1983 with a degree in computer science. Upon graduation, he moved to Las Vegas, Nevada, to begin work as a VAX/VMS system manager for the United States Air Force. In 1986, he received his master's degree in computer science from the University of Nevada, Las Vegas. Kris taught computer science at National University for one year before leaving the Air Force in 1988 to begin writing full-time.

DOS: The Complete Reference,
Fourth Edition

Kris Jamsa

Osborne **McGraw-Hill**

Berkeley New York St. Louis San Francisco
Auckland Bogotá Hamburg London Madrid
Mexico City Milan Montreal New Delhi Panama City
Paris São Paulo Singapore Sydney
Tokyo Toronto

Osborne **McGraw-Hill**
2600 Tenth Street
Berkeley, California 94710
U.S.A.

Osborne **McGraw-Hill** offers software for sale. For information on software, translations, or book distributors outside of the U.S.A., please write to Osborne **McGraw-Hill** at the above address.

DOS: The Complete Reference, Fourth Edition

Copyright © 1993 by McGraw-Hill, Inc. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

90 DOC 99

ISBN 0-07-881904-0

Information has been obtained by Osborne **McGraw-Hill** from sources believed to be reliable. However, because of the possibility of human or mechanical error by our sources, Osborne **McGraw-Hill**, or others, Osborne **McGraw-Hill** does not guarantee the accuracy, adequacy, or completeness of any information and is not responsible for any errors or omissions or the results obtained from use of such information.

To Debbie,
Most people only dream at night,
with you my dreams last a lifetime...

Acquisitions Editor
Jeffrey M. Pepper

Associate Editor
Vickie Van Ausdall

Technical Editor
Phillip Schmauder

Project Editor
Janet Walden

Copy Editor
Paul Medoff

Proofreader
Linda Medoff

Indexer
Phillip Schmauder

Computer Designer
Fred Lass

Illustrators
Susie C. Kim
Peter F. Hancik

Cover Designers
Bay Graphics Design, Inc.
Mason Fong

Contents at a Glance

1 Getting to Know Your Computer 1
2 Getting DOS Started 31
3 Issuing DOS Commands 43
4 Getting Started with DOS Files 63
5 Understanding Your Keyboard 85
6 Getting Around Your Disk 105
7 Preparing Disks for Use 117
8 Fundamental File Operations 137
9 DOS Wildcard Characters 153
10 Directories 171
11 Getting the Most from DOS Directories 193
12 Fundamental Copy Operations 217
13 Printing 229
14 Getting the Most from PRINT 243
15 Grouping Commands with Batch Files 259
16 Decision-Making Batch Files 279
17 Getting the Most from Batch Files 303
18 Automatic System Startup Commands with AUTOEXEC.BAT 321
19 The DOS Environment 333
20 The DOS Redirection Operators 347
21 The DOS Pipe 367
22 Protecting Your Files Using Backups 383

23 Advanced File Manipulation 417
24 Customizing Your System with CONFIG.SYS 443
25 Using CHKDSK 473
26 DOS Pretender Commands 491
27 DOS 5 and 6 DOSKEY and Macros 507
28 DOS 5 and 6 Crisis Prevention 521
29 Using DEBUG 553
30 Another Look at Disks 583
31 Improving Your System Performance 599
32 Understanding DOS Memory Management 613
33 How DOS Actually Stores Files 647
34 ANSI.SYS 659
35 Advanced Disk Topics 679
36 Linking Computers with DOS 6 689
37 The DOS 5 and 6 Editor, EDIT 699
38 Support for International Users 715
39 Using the DOS 5 and 6 Shell 733
40 Advanced DOS 5 and 6 Shell Techniques 763
41 DOS System Commands 775
42 Detecting and Correcting Disk Fragmentation 815
43 Disk Partitions and FDISK 831
44 Doubling Disk Storage Capacity with DOS 6 845
45 The History of DOS 857

Command Reference 865

A Exit Status Values 1059
B ASCII and Extended ASCII Character Sets 1065
C International Keyboard Templates and Code Pages 1069
D Upgrading to DOS 6 1083
E Common Error Messages 1089

Index 1099

Contents

Acknowledgments xix
Introduction xxi

1

Getting to Know Your Computer 1

Looking at Your Computer's Hardware 1
Connecting the Pieces 3
How Disks Store Information 6
Floppy Disks in Detail 8
Looking at Disk Drives 14
Understanding Software 16
Understanding DOS 18
Hands On 20
Review 20
Answers 22
Advanced Concepts 24
Key Terms 30

2

Getting DOS Started 31

Before You Begin 31
Starting DOS from Floppy Disks 32
Turning On Your Computer 32
What DOS Will Display 33
Troubleshooting the DOS Startup 36
Hands On 37
Review 37
Answers 38
Advanced Concepts 38
Key Terms 42

3

Issuing DOS Commands 43

Executing Your First DOS Commands 43
Command-Line Parameters 50
External Versus Internal Commands 51
The "Bad command or file name" Message 52
Online Help for DOS 6 Commands 55
Hands On 58
Review 59
Answers 59
Advanced Concepts 60
Key Terms 61

4

Getting Started with DOS Files 63

A Directory Is a List of Related Files 63
Using the DIR Command 64
Hands On 74

Review 75
Answers 76
Advanced Concepts 77
Key Terms 83

5

Understanding Your Keyboard 85

Keyboards Have Four Major Sections 85
Key Combinations 92
Hands On 95
Review 97
Answers 98
Advanced Concepts 98
Key Terms 103

6

Getting Around Your Disk 105

Changing the Default Drive 105
Accessing a Drive's Disk Without Changing Default Drives 106
The Abort, Retry, Fail? Prompt 108
Hands On 111
Review 112
Answers 113
Advanced Concepts 114
Key Terms 116

7

Preparing Disks for Use 117

FORMAT Is an External Command 117
The Formatting Process 118
Not All Disk Sectors Are Capable of Storing Information 120
Preparing a Floppy Disk for Use 121
Formatting a Disk with Less Storage Capacity than the Floppy Drive 123
Formatting Your Hard Disk 123
Hands On 125
Review 126
Answers 126
Advanced Concepts 127
Key Terms 135

8

Fundamental File Operations 137

TYPE Displays a File's Contents 137
COPY Duplicates a File's Contents 139
RENAME Changes a File's Name 143
DEL Deletes Files 144
Hands On 147
Review 148
Answers 149
Advanced Concepts 149
Key Terms 152

9

DOS Wildcard Characters 153

The * Wildcard 153
The ? Wildcard 161
Hands On 164
Review 166
Answers 167
Advanced Concepts 167
Key Term 169

10

Directories 171

The Root Directory 171
The Current Directory 172
CHDIR Changes the Current Directory 173
MKDIR Creates Directories 176
RMDIR Removes Directories 181
Hands On 183
Review 185
Answers 186
Advanced Concepts 187
Key Terms 191

11

Getting the Most from DOS Directories 193

Displaying Your Directory Tree 193
Simplifying Execution of External Commands 197
Abbreviating Directory Names 202
FASTOPEN Improves System Performance 205
Hands On 209
Review 210
Answers 211
Advanced Concepts 212
Key Terms 215

12

Fundamental Copy Operations 217

Copying Files from a Hard Disk to a Floppy Disk 217
Copying from a Floppy Disk to a Hard Disk 218
DISKCOPY Copies an Entire Disk's Contents 219
Hands On 224
Review 225
Answers 225
Advanced Concepts 226
Key Terms 227

13

Printing 229

Using Your Printer 229
Identifying Your Printer Type 230
The Device Names PRN and AUX 231
The Serial Port's Communication Settings 231
Copying Files to Your Printer 232
Printing the Screen 233
Sending Information to Both Screen and Printer 236
Telling Your Applications About Your Printer 236
Hands On 236
Review 239
Answers 239
Advanced Concepts 239
Key Terms 241

14

Getting the Most from PRINT 243

Understanding PRINT 243
Hands On 249
Review 250
Answers 251
Advanced Concepts 252
Key Terms 257

15

Grouping Commands with Batch Files 259

Creating Simple Batch Files 259
Three Commonly Used Batch Commands 265
Hands On 270
Review 272
Answers 272
Advanced Concepts 273
Using Extended ASCII Characters 275
Key Terms 277

16

Decision-Making Batch Files 279

The IF Command 280
IF EXIST Tests for Files 280
Using IF to Compare Two Character Strings 287
Passing Information to Your Batch Files 287
Hands On 292
Review 293
Answers 294
Advanced Concepts 294
Using the DOS 6 CHOICE

Command 297
Key Terms 301

17

Getting the Most from Batch Files 303

Branching from One Location to Another Using GOTO 303
Repeating a Command for a Set of Files 306
Accessing More than Nine Batch Parameters 308
Invoking One Batch File from Within Another 311
Hands On 314
Review 316
Answers 317
Advanced Concepts 317
Key Terms 320

18

Automatic System Startup Commands with AUTOEXEC.BAT 321

Common AUTOEXEC.BAT Commands 322
A Sample AUTOEXEC.BAT 326
AUTOEXEC.BAT and CONFIG.SYS 327
Change AUTOEXEC.BAT with Caution 327
Hands On 328
Review 329
Answers 329
Advanced Concepts 330
Key Terms 331

19

The DOS Environment 333

The SET Command 333
How DOS Uses the Environment 336
Hands On 339
Review 340
Answers 341
Advanced Concepts 341
Key Terms 346

20

The DOS Redirection Operators 347

The Output Redirection Operator 347
The Append Redirection Operator 350
Possible Problems When Redirecting Output 351
The Input Redirection Operator 352
Combining Input and Output Operators 356
Hands On 358
Review 359
Answers 359
Advanced Concepts 360
Key Terms 365

21

The DOS Pipe 367

Locating the Pipe Operator on Your Keyboard 367
Understanding the DOS Pipe 368
Which Commands Can You Pipe? 370
Combining the Pipe and I/O Redirection Operators 371
Pipe Operator Pitfalls 372
Hands On 374
Review 376
Answers 376
Advanced Concepts 378
Key Terms 381

22

Protecting Your Files with Backups 383

Understanding the Backup Process 384
Backing Up Your Entire Hard Disk 386
Performing Daily Backup Operations 388
Using BACKUP to Back Up a Key File or Directory 391
Restoring Backup Files to Your Hard Disk 391
Restoring a Specific File or Directory 392
Using Third-Party Backup Utility Programs 394
Using the DOS 6 MSBACKUP Command 395
Using MSBACKUP 397

Hands On 407
Review 409
Answers 409
Advanced Concepts: BACKUP and RESTORE 410
Advanced Concepts: MSBACKUP 414
Key Terms 416

23

Advanced File Manipulation 417

Manipulating File Attributes with ATTRIB 417
XCOPY's Extended Copy Capabilities 421
Comparing Two Files with COMP 424
Comparing Files with FC 429
Adding and Replacing Selected Files with REPLACE 431
Hands On 435
Review 439
Answers 439
Advanced Concepts 440
Key Terms 441

24

Customizing Your System with CONFIG.SYS 443

Understanding the CONFIG.SYS Entries 443
Hands On 454
Review 455
Answers 455
Advanced Concepts 456
DOS 6 System Configuration Options 464
Key Terms 472

25

Using CHKDSK 473

Understanding CHKDSK 473
Using CHKDSK to Correct Disk Errors 480
Handling Other CHKDSK Errors 484
Using CHKDSK to Display Filenames 484
Hands On 485
Review 486
Answers 486
Advanced Concepts 487
Key Terms 489

26

DOS Pretender Commands 491

Disk Drive Identity Swapping with ASSIGN 491
Substituting a Drive Letter for a Directory Name with SUBST 493
Making DOS Think a Drive Is a Directory with JOIN 496
Hands On 498
Review 502
Answers 503
Advanced Concepts 504
Key Terms 504

27

DOS 5 and 6 DOSKEY and Macros 507

Using DOSKEY to Recall Commands 507
Hands On 513
Review 514
Answers 514
Advanced Concepts 514
Key Terms 520

28

DOS 5 and 6 Crisis Prevention 521

Retrieving Deleted Files with UNDELETE 521
Protecting Your Disk and Files with File Tracking 530
Recovering a Disk with UNFORMAT 536
Safe Versus Unsafe Format Operations 543
Hands On 545
Review 547
Answers 548
Advanced Concepts 549
Key Terms 552

29

Using DEBUG 553

Creating Simple COM Commands 553
Hands On 559
Review 562
Answers 563
Advanced Concepts 565
Key Terms 582

30

Another Look at Disks 583

Disk Volume Labels: LABEL and VOL 583
Comparing Two Floppy Disks with DISKCOMP 588
File Sharing and Large Disk Support with SHARE 589
Verifying Disk Writes with VERIFY 590
Salvaging a File or Disk with RECOVER 591
Hands On 595
Review 596
Answers 596
Advanced Concepts 597
Key Terms 598

31

Improving Your System Performance 599

Directory Considerations for Maximum Performance 599
File Considerations for Maximum Performance 601
Disk Considerations for Maximum Performance 602
Creating a Disk Cache 603
System Considerations for Performance 606
Hands On 608
Review 609
Answers 610
Advanced Concepts 610
Key Terms 612

32

Understanding DOS Memory Management 613

Understanding Memory Types 613
How We Use Extended and Expanded Memory 621
Viewing Your Memory Usage 623
DOS 5 or later Memory Management 627
Using the DOS 6 MEMMAKER Command 631
Hands On 636
Review 639
Answers 639
Advanced Concepts 640
Key Terms 644

33

How DOS Actually Stores Files 647

The File Allocation Table 647
Directory Entries 650
DOS Stores the FAT and Root Directory at Fixed Locations 652
How FORMAT Works 653
How CHKDSK Works 654
Review 656
Answers 656
Advanced Concepts 658
DOS 6 DBLSPACE Command 658
Key Terms 658

34

ANSI.SYS 659

Installing ANSI.SYS 659
Escape Sequences 659
Controlling Screen Colors Using ANSI.SYS 663
Redefining Keys with ANSI.SYS 665
Creating ANSI.COM to Execute Escape Sequences 669
Hands On 672
Review 672
Answers 673
Advanced Concepts 673
Key Term 677

35

Advanced Disk Topics 679

Understanding Disk Head Parking 679
Disk Sector Interleaving 682
Low-Level Formatting 684
Review 685
Answers 686
Advanced Concepts 687
Key Terms 687

36

Linking Computers with DOS 6 689

Understanding Clients and Servers 689
Hands On 697
Review 697
Answers 697
Key Terms 698

37

The DOS 5 and 6 Editor, EDIT 699

Starting EDIT 699
Hands On 709
Review 710
Answers 710
Advanced Concepts 711
Key Terms 714

38

Support for International Users 715

Selecting a Code Page 715
Setting an International Configuration 716
Selecting a Symbol Set 718
Selecting a Keyboard Template 720
Preparing Code Pages for Use 723
Changing Code Pages with CHCP 726
Hands On 729
Review 730
Answers 730
Advanced Concepts 731
Key Terms 731

39

Using the DOS 5 and 6 Shell 733

Starting the DOS 5 and 6 Shell 733
Displaying the Shell in Graphics Mode 737
Working with Directory Trees 737
Exiting the DOS Shell 738
Working with Files 738
Changing the Shell Display 750
Running Programs from the Program List 752
Using the Shell's Online Help 756
Hands On 758
Review 759
Answers 759
Advanced Concepts 760
Key Terms 762

40

Advanced DOS 5 and 6 Shell Techniques 763

Setting Your Screen Colors 763
Working with Program Groups 764
Working with Program Entries 767
Hands On 770
Review 771
Answers 771
Advanced Concepts 772
Key Terms 774

41

DOS System Commands 775

Customizing Your System Prompt 775
Enabling Extended CONTROL-BREAK Checking 778
Using EXIT to Resume a Program 780
Online Help for DOS 5 Commands 780
Online Help for DOS 6 Commands 782
Displaying Extended ASCII Characters with GRAFTABL 784
Invoking the BASIC Programming Language 785
Revisiting MODE 786

DOS 6 Advanced Power Management 794
Hands On 798
Review 799
Answers 799
Advanced Concepts 800
Using MSD, Microsoft's Diagnostic Software 806
Key Terms 814

42

Detecting and Correcting Disk Fragmentation 815

Understanding Fragmented Files 815
Correcting File Fragmentation Prior to DOS 6 820
Defragmenting Files Using DOS 6 822
Fragmented Files Can Prevent Successful File Undeletion 826
Hands On 828
Review 828
Answers 828
Advanced Concepts 829
Key Terms 830

43

Disk Partitions and FDISK 831

Understanding Hard Disk Partitions 831
Using FDISK 834
Creating a Primary and an Extended Partition 837
Hands On 841
Review 841
Answers 842
Advanced Concepts 842
Key Terms 844

44

Doubling Your Disk's Storage Capacity 845

Doubling Your Hard Disk's Storage Capacity 845
Hands On 851
Review 852
Answers 852
Advanced Concepts 853
Key Terms 855

45

The History of DOS 857

Before DOS 857
DOS 1 858
DOS 1.1 859
DOS 2 859
DOS 3 860
DOS 4 861
DOS 5 861
DOS 6 862
Conclusion 863

***Command Reference* 865**

APPEND 865
ASSIGN 867
ATTRIB 869
BACKUP 871
BREAK 874
CALL 876
CHCP 876
CHDIR 878
CHKDSK 879
CHOICE 885
CLS 887
COMMAND 887
COMP 889
COPY 894
CTTY 896
DATE 897
DBLSPACE 899
DEFRAG 902
DEL 904
DELTREE 906
DIR 908
DISKCOMP 911
DISKCOPY 915
DOSKEY 919
DOSSHELL 922
ECHO 923
EDIT 925
EXIT 926
EXPAND 926
FASTHELP 927
FASTOPEN 929
FC 931
FDISK 935
FIND 941

FOR 944
FORMAT 945
GOTO 952
GRAFTABL 953
GRAPHICS 955
HELP 957
IF 959
INTERLNK 961
INTERSVR 963
JOIN 965
KEYB 966
LABEL 970
LOADFIX 973
LOADHIGH 974
MEM 975
MEMMAKER 977
MIRROR 979
MKDIR 982
MODE 984
MORE 991
MOVE 992
MSAV 994
MSBACKUP 996
MSD 997
NLSFUNC 999
PATH 1000
PAUSE 1001
POWER 1003
PRINT 1004
PROMPT 1008
RECOVER 1009
REM 1011
RENAME 1013
REPLACE 1014
RESTORE 1018
RMDIR 1022
SET 1023
SETVER 1025
SHARE 1028
SHIFT 1029
SMARTDRV 1030
SORT 1033
SUBST 1035
SYS 1037
TIME 1039
TREE 1040
TYPE 1042
UNDELETE 1043
UNFORMAT 1046
VER 1048
VERIFY 1049
VOL 1051
VSAFE 1052
XCOPY 1054

A

Exit Status Values 1059
BACKUP 1059
DEFRAG 1059
DISKCOMP 1060
DISKCOPY 1060
FORMAT 1060
GRAFTABL 1061
KEYB 1061
REPLACE 1061
RESTORE 1062
SETVER 1062
XCOPY 1063

B

ASCII and Extended ASCII Character Sets 1065

C

International Keyboard Templates and Code Pages 1069

D

Upgrading to DOS 6 1083
Performing the DOS 6 Upgrade 1083
Using the Uninstall Disks 1087

E

Common Error Messages 1089

Index 1099

Acknowledgments

When Osborne and I began this update, we knew that we were facing a large and difficult project. In its first three editions, *DOS: The Complete Reference* had become the resource readers knew they could turn to for answers on any question about DOS. With the release of DOS 6, and the fourth edition of this book, our goal was to ensure that we provided readers with the quality they had come to expect. To achieve that goal, many people worked endless days to make sure we met your expectations. Please take a few moments to review the list of individuals (which appears at the front of this book) who unselfishly gave of themselves and fought to "do things right." Their patience and dedication is sincerely appreciated. As your read through the chapters of this book, I am sure you will agree that we met our goal.

Introduction

Since DOS was first introduced in 1981, many books on DOS have flooded the bookstores, making it difficult for users to determine which DOS book is right for them. For the past twelve years, I have been studying DOS and have written over a dozen books on DOS, from getting started with DOS, to advanced discussions on its inner workings. *DOS: The Complete Reference, Fourth Edition,* contains the facts, tips, and secrets I have collected on DOS since its release in 1981. It is the most thorough discussion on DOS available.

What DOS Versions are Discussed?

When software developers make changes to a program and release the updated program to users, they assign a version number to the software. Since its release in 1981, DOS has grown from version 1.0 to the recently released DOS 6.0. Along the way, the number of people using DOS has grown beyond 90 million.

DOS: The Complete Reference, Fourth Edition, discusses the commands used with every DOS version. Regardless of the DOS version you are using, this book contains the answers to your questions.

Who Should Use This Book?

There are over 90 million DOS users and no two have the same level of understanding. With that fact in mind, this book presents each topic beginning with the basics and then lets you advance into detailed discussion on a topic as you feel comfortable.

New DOS Users

Don't be intimidated by this book's size. Each chapter goes to great lengths to explain new concepts in a straightforward manner. I have included many illustrations to help you understand the topics easily, all of which contribute to the book's size.

Because all new DOS users have different backgrounds, each chapter begins with a new topic, and then explains how and when you would use it, as well as giving some helpful hints. A "Hands On" section (that lets you perform commands) follows the introductory discussion. In this way, you can't hurt your computer, or lose information it contains. Every chapter contains a "Review" section with a few questions on concepts I consider most valuable. (Don't worry, the "Answer" section follows the questions.) Use these questions, along with the handy "Summary" sections found in most chapters, as your guide to the key points you should retain from the chapter.

Intermediate DOS Users

If you have worked with DOS for a while, this book will take you to the power-user status. I have included the tricks and shortcuts you can use to make your time with the computer more productive. Several of the book's introductory chapters may be "old hat" to you by now. Read each of these chapter's "Summary" section and make sure you can answer corresponding questions. If a topic is new to you, make sure you read the discussion on the topic within the chapter. Also, skim each chapter's "Advanced Concepts" section. Don't worry if you don't thoroughly understand the discussion. By the time you complete this book's latter chapters, you will have the foundation you need to master the advanced concepts.

Advanced DOS Users

If you are looking for a single book to which you can turn for all your DOS questions, you have the right book in your hands. In addition, you will find many more secrets and advanced techniques in each chapter's "Advanced Concepts" section. Much of this discussion will teach you the steps DOS performs behind the scenes, making your knowledge of DOS complete.

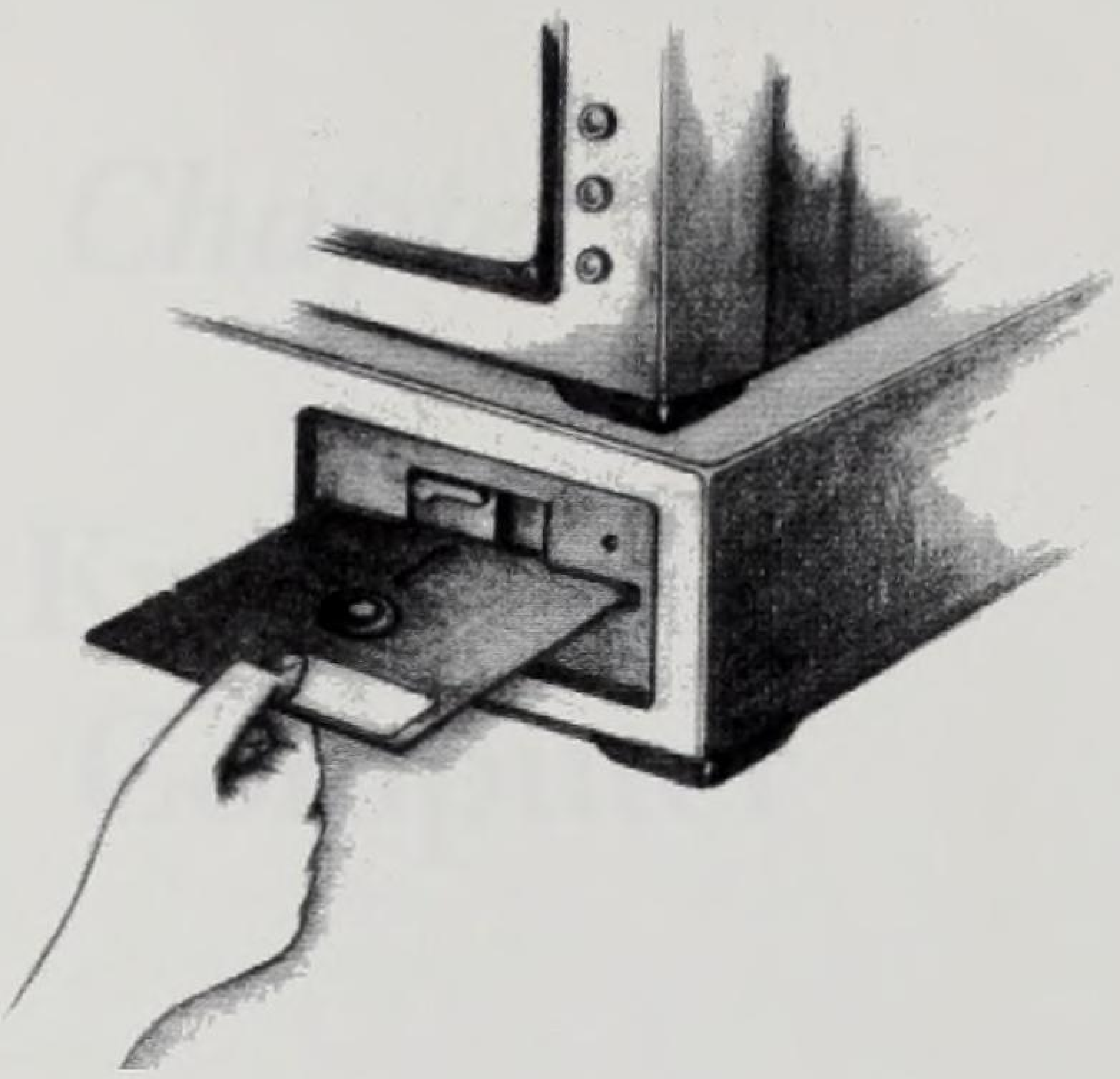
Companion Diskette Packages

As you work with DOS on a regular basis, there will be times you need details on a specific command or concept. To complement this book's text, we have made two diskette packages available. The *DOS HELP* diskette package provides an on-line discussion of every DOS command, the command's format, tips, and examples of the command's use. *DOS HELP* places the answers to your DOS questions right at your fingertips.

DOS: The Complete Reference Companion Diskette contains all DEBUG script routines presented throughout this book, along with a collection of utility programs that you can use with the DOS pipe and I/O redirection operators, and within your DOS batch files.

If you are interested in either disk, see the order form following this Introduction.

Special Offer



One of the greatest frustrations users experience when working with DOS is forgetting a command. ***DOS HELP*** is a software program you can copy to your hard disk; it contains all of the answers to your DOS questions.

DOS HELP displays a friendly menu that summarizes all the DOS commands. By highlighting a command with your arrow keys and pressing ENTER, you can discover command specifics, view common examples of the command in use, and read tips that make you even more effective.

Every computer user should have a copy of ***DOS HELP***!

As you examine different concepts throughout this book, you will run into cases where the commands DOS provides don't give you all the capabilities you need. To help you better use your computer, we offer a disk containing several commands that DOS forgot. ***DOS: The Complete Reference Companion Disk*** contains the batch and DEBUG script files presented in this book, as well as several useful commands. Table 21-1 in Chapter 21 lists several of the commands you will find.

Ordering Information

Domestic Orders:
For *DOS HELP*, please send $24.95 (includes shipping and handling) to the address below. For fastest service, please send a money order or cashier's check. Please allow 2 to 3 weeks for delivery.

For *Companion Disk*, please send $9.95 (includes shipping and handling).

Foreign Orders:
For *DOS HELP*, please send $29.95 (USD; includes shipping and handling) to the address below. For fastest service, please send an international money order. Please allow 4 to 6 weeks for delivery.

For *Companion Disk*, please send $12.95 (includes shipping and handling).

(Credit card orders not accepted)

Name: ____________________
Address: ____________________
City: ____________________ State: ______ ZIP: __________
Country: ____________________ Phone: __________
Please send: ______ copies of *DOS HELP* $__________
______ copies of *Companion Disk* $__________
Enclosed is my check or money order for $__________

Please send your order with payment to:
Concept Software
P.O Box 26981
Las Vegas, NV 89126

This is solely the offer of the author. Osborne/McGraw-Hill takes no responsibility for the fulfillment of this offer.

Chapter 1

Getting to Know Your Computer

Computers are incredibly powerful machines that affect almost every aspect of our lives. Computers work behind the scenes to control stoplights, to handle the millions of phone calls made each day, to control aircraft, and even to control the microwave oven you may have used to cook dinner. In fact, computers are so common that most of us take them for granted—until, that is, someone puts a personal computer on your desk.

This chapter takes a look at the different pieces that make up your computer. If you've never worked with a personal computer (PC for short) before, don't worry. Chapter 1 starts you off at square 1, and teaches you the fundamentals you'll use as the base for your learning throughout the remainder of this book.

Looking at Your Computer's Hardware

Your computer's hardware consists of its physical parts such as the keyboard, screen, printer, chassis, and, perhaps, mouse. Figure 1-1 illustrates the hardware parts common to most computers today.

In general, the *keyboard* lets you type commands, letters, reports, or numbers. If you don't know how to type, don't worry. Typing on a computer's keyboard is much easier than using a typewriter, because you can quickly correct mistakes.

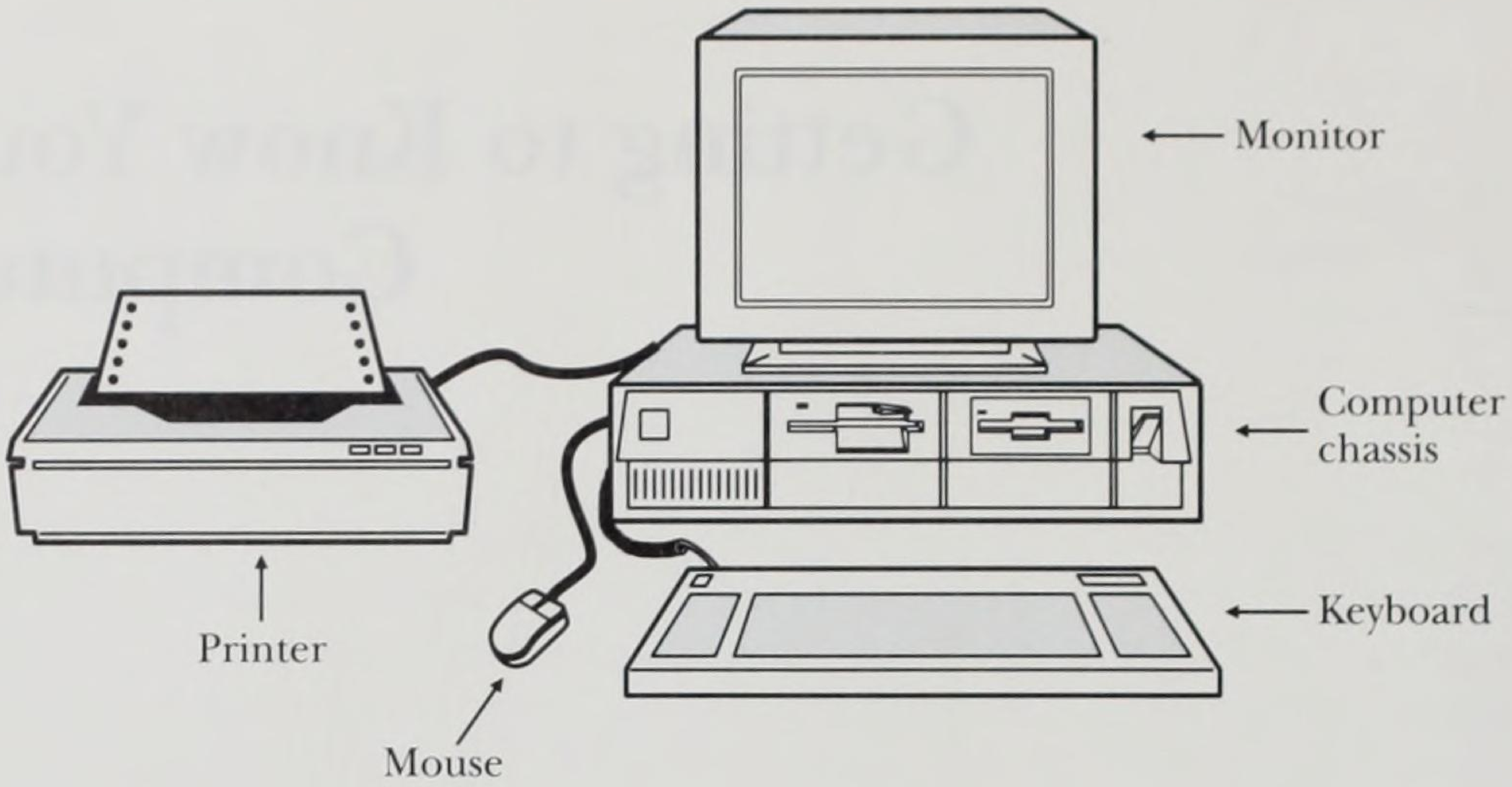


Figure 1-1. *Common hardware parts*

Your computer's *screen* (often called a monitor) displays a program's result, such as a graph, or the information you are currently typing, possibly a report or memo. Your computer screen works very much like a television, with its own on/off switch and brightness controls.

The *printer* lets you make printed copies of your letters, reports, spreadsheets, or even graphs of information. Like your monitor, the printer has its own on/off switch.

Your computer's *chassis* houses your computer's key electronic parts, such as its electronic memory and the microprocessor (often called the *CPU* or *central processing unit*). In addition, the chassis holds your computer's disk drives, which let you store information from one session at the computer to the next. As you will learn later in this chapter, most computers use two types of disks. A *hard disk* is a very fast nonremovable disk that can store very large amounts of information. *Floppy disks,* on the other hand, are slower, removable disks with less storage capacity. Floppy disks allow you to exchange information between computers or load new programs on your hard disk.

Last, a *mouse* is a hardware device that simplifies your use of many programs, such as Microsoft Windows or the DOS shell provided with DOS version 4 or later. Using the mouse, you can often aim a mouse pointer, which appears on your computer screen, at different menu options to select them.

Table 1-1 briefly summarizes common hardware devices.

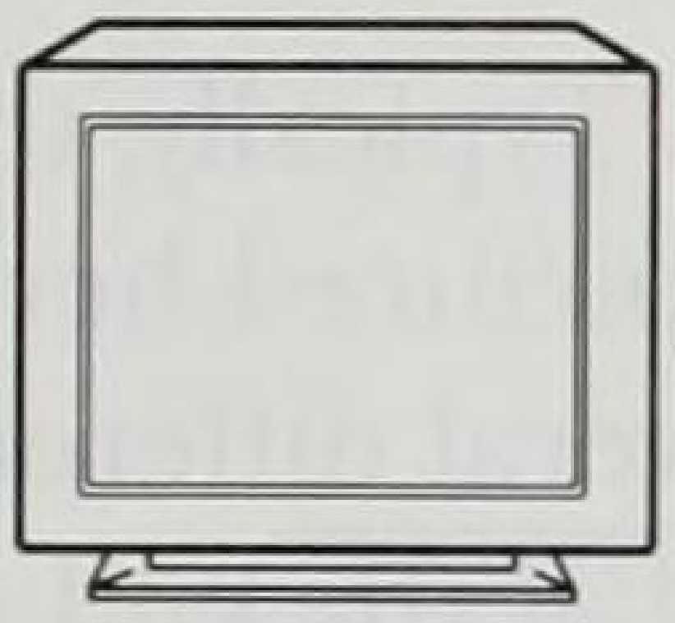
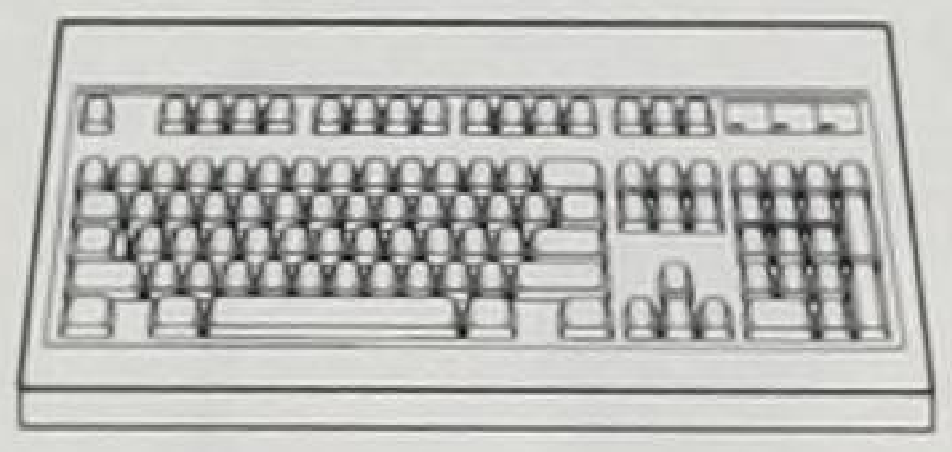
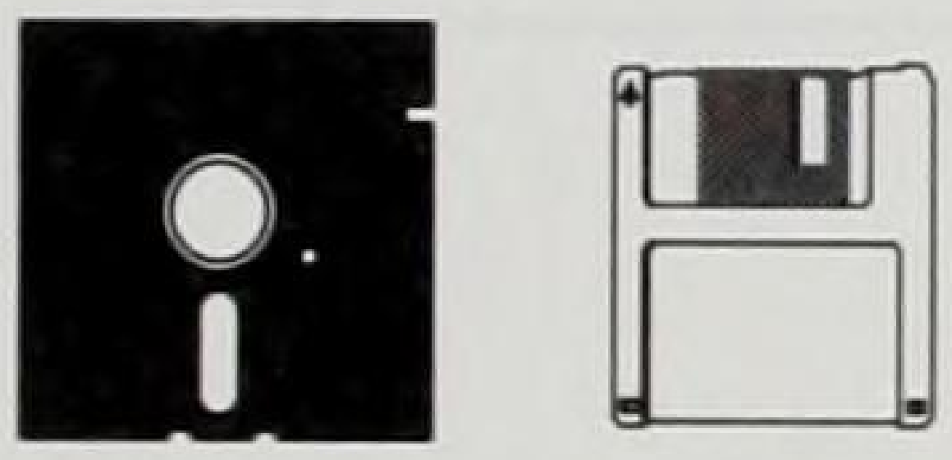
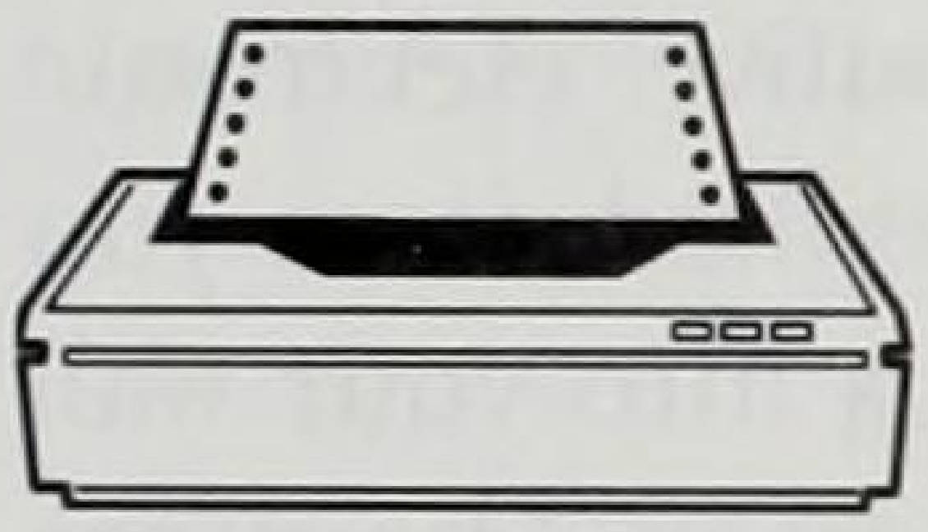
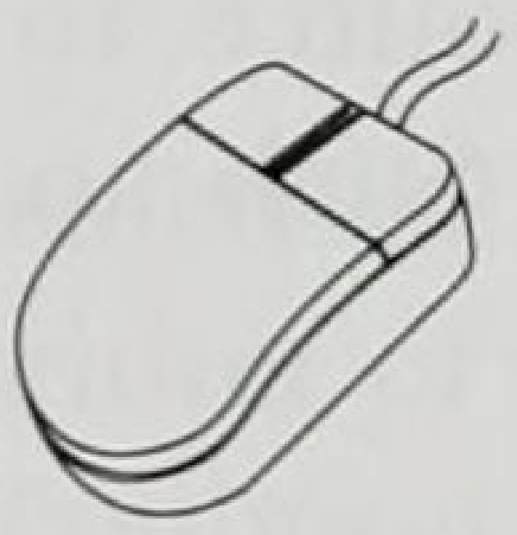
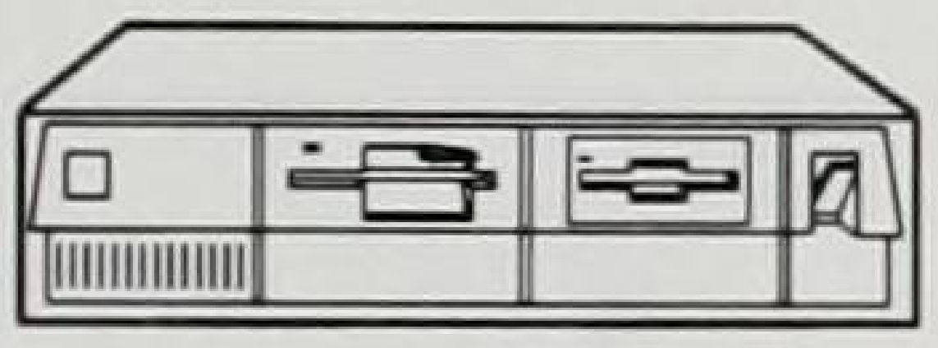
Component	**Name**	**Function**
	Screen	Lets programs display meaningful results to you and lets you view the information you type at the keyboard
	Keyboard	Lets you respond to a program's prompt for information and lets you type in information for storage or processing
	Disk	Lets you store information from one computer session to the next
	Printer	Lets you obtain a printed copy of a program's result
	Mouse	Lets you move the cursor or pointer easily; lets you choose menu options or files by clicking on them
	Chassis	Houses your computer's central processing unit, memory, and other hardware boards, as well as your computer's floppy and hard disk drives

Table 1-1. *Common Hardware Components and Functions*

Connecting the Pieces

Most new users are introduced to a computer that is already up and running. If you buy a new computer, however, you will very likely have to set up the computer yourself. As you will learn in this section, setting up a computer is actually quite easy. In general, you will connect your keyboard, monitor, and printer to your computer's chassis, and then individually plug the chassis, monitor, and printer into a wall outlet.

To begin, your keyboard typically plugs into the back of your computer's chassis, as shown in Figure 1-2. Examine the keyboard's plug closely to make sure the pins line up with the holes in the chassis.

In a similar way, your monitor will plug into the back of the chassis. Typically, the back side of the chassis (back-right side of the common chassis style pictured here, and back-top side of the upright "tower" chassis style) will contain several different device sockets called *ports*. Compare each port's connector to the plug at the end of your video cable, as shown in Figure 1-3.

In this same way, you must connect your printer cable to the back of your computer chassis, as shown in Figure 1-4.

Last, after you plug your chassis, monitor, and printer and plug the AC power cord into a wall outlet, your system is ready to go!

Using a Surge Suppressor

As you can imagine, your computer contains thousands of sensitive electronic components. To protect your computer, a critical, yet often overlooked, item you should buy and use is a surge suppressor. A surge suppressor plugs into your wall outlet, and your computer equipment in turn plugs into the suppressor. The surge suppressor sits between your computer equipment and the electrical source to prevent a very large surge of electricity from damaging your computer. Such an

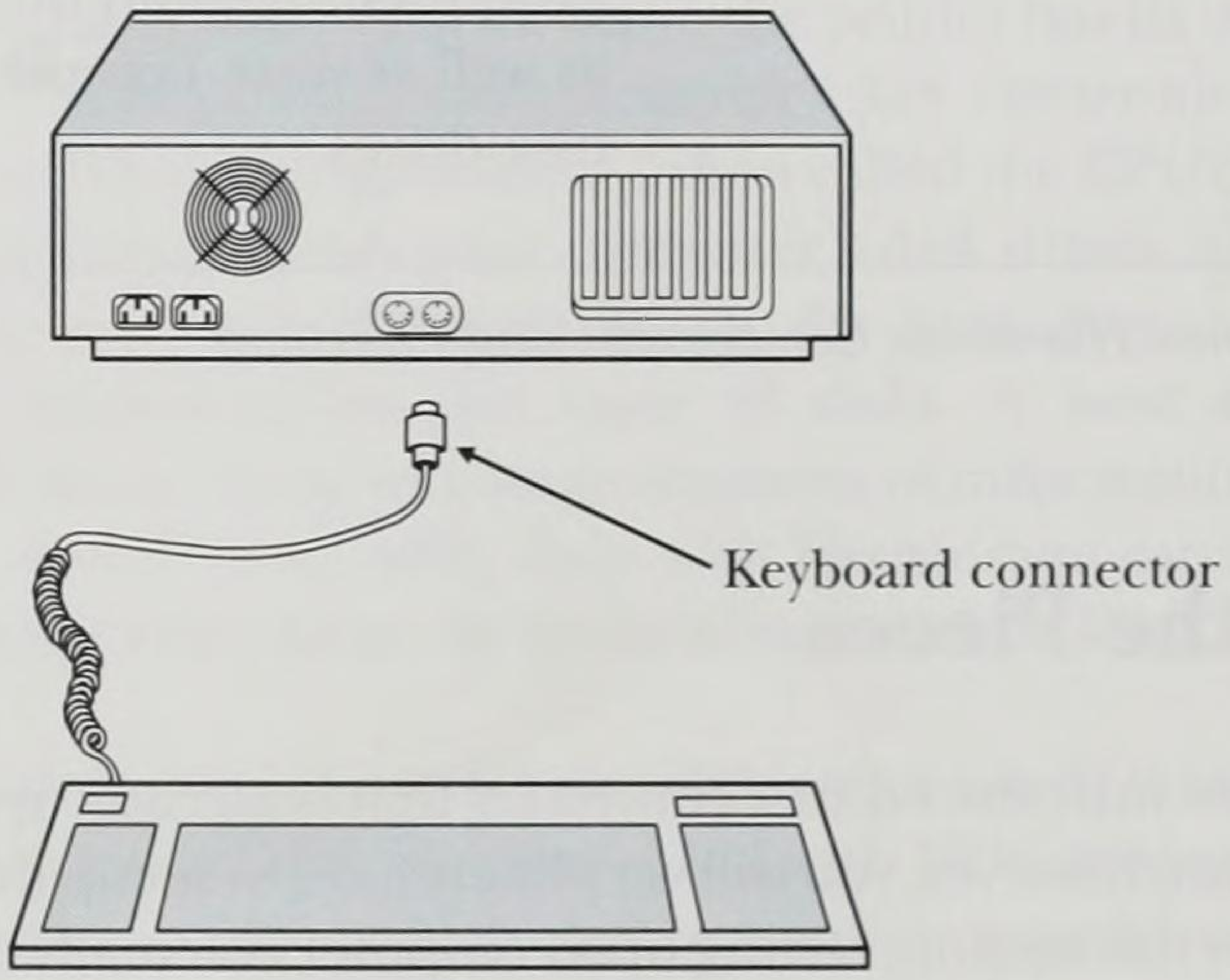


Figure 1-2. Keyboards typically plug into the back of your computer's chassis

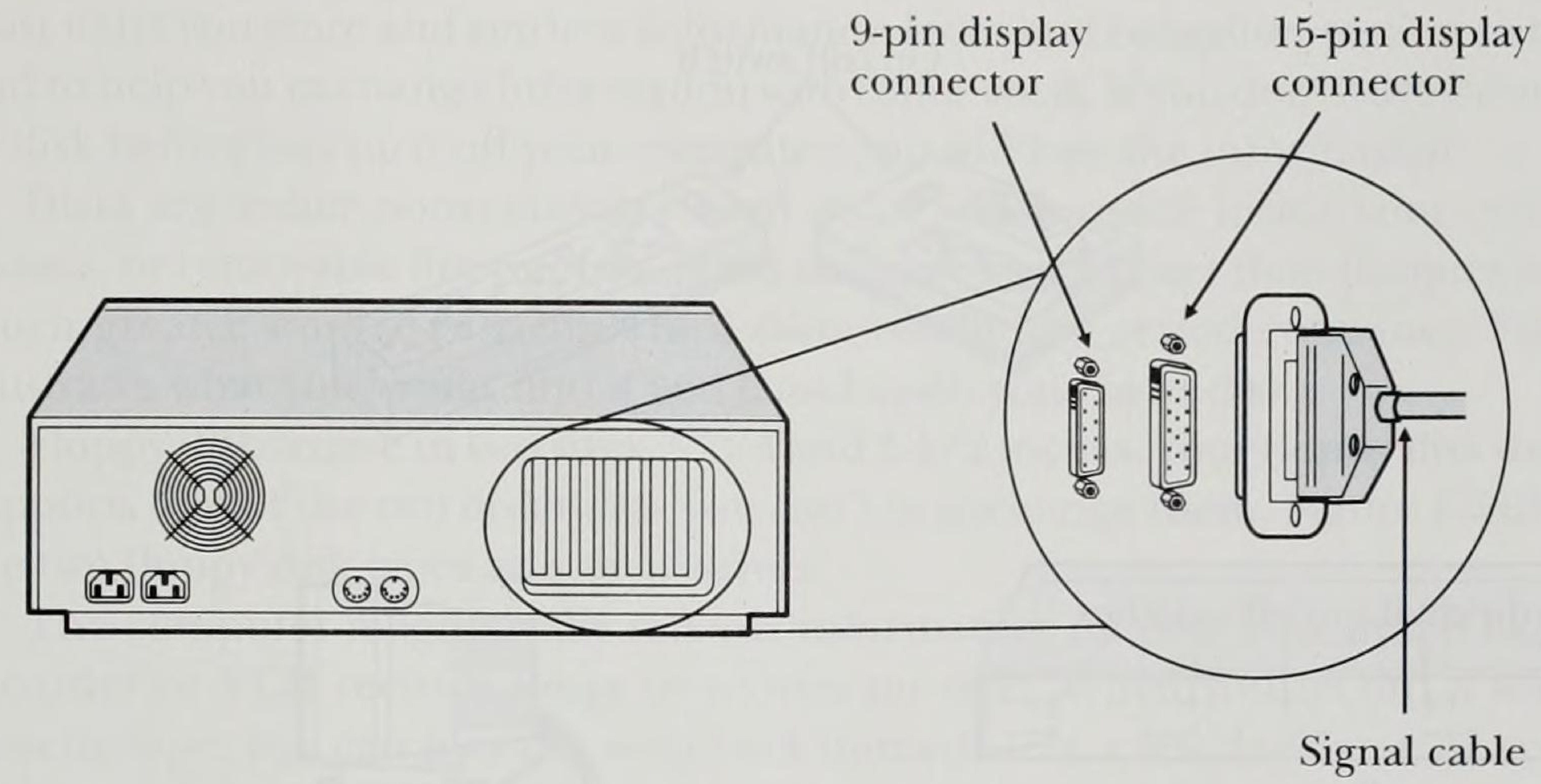

9-pin display connector
15-pin display connector
Signal cable

Figure 1-3. *Match your video plug to one of the ports at the back of your computer (9-pin for CGA or EGA monitor, or 15-pin for VGA monitor)*

electrical surge can be caused by lightning or even a faulty electrical transformer, either of which can occur several miles from your computer. Figure 1-5 shows three common types of surge suppressors.

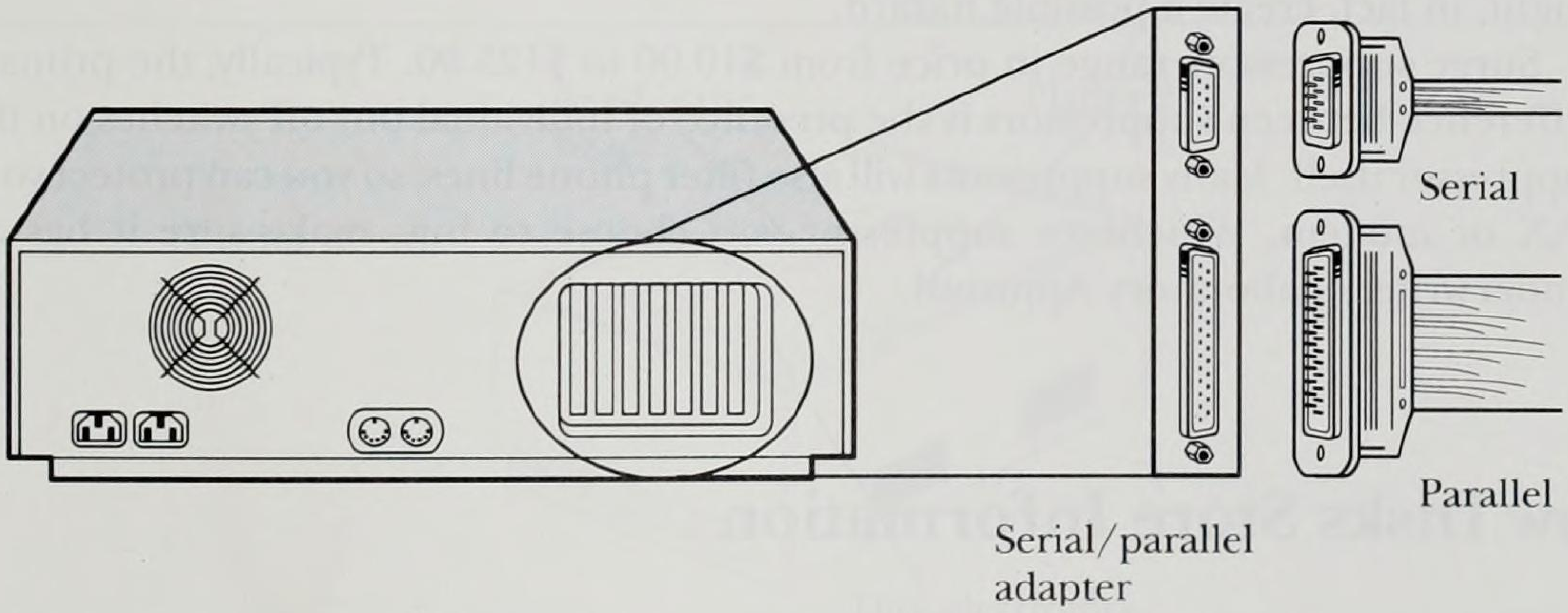

Serial
Parallel
Serial/parallel adapter

Figure 1-4. *Match your printer cable to one of the ports*

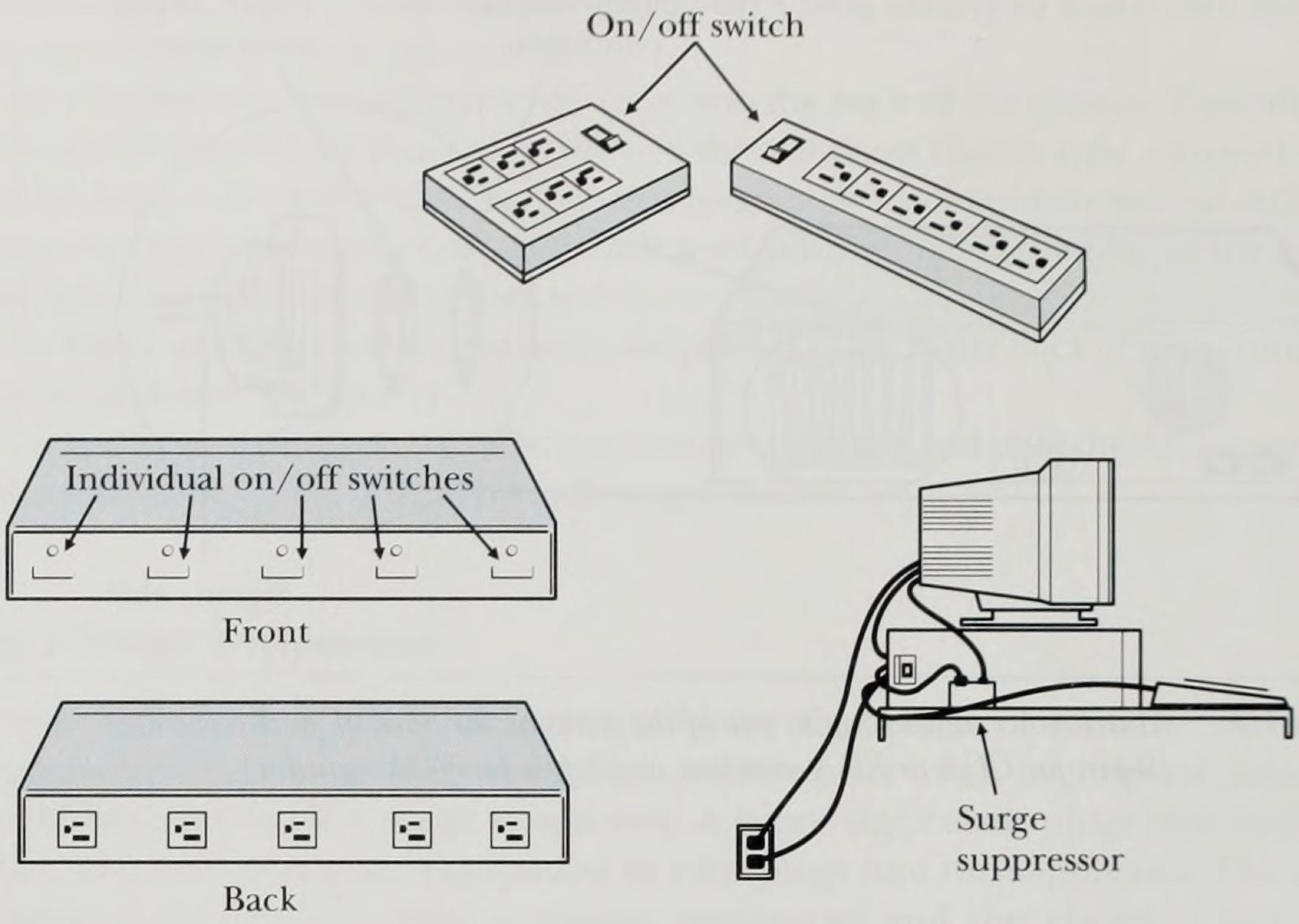


Figure 1-5. *Common types of surge suppressors*

Do not confuse surge suppressors with power strips that simply let you plug several devices into one outlet. A power strip does not protect the devices in any way, but might, in fact, create a possible hazard.

Surge suppressors range in price from $10.00 to $125.00. Typically, the primary difference between suppressors is the presence of individual on/off switches on the suppressor itself. Many suppressors will also filter phone lines, so you can protect your FAX or modem. Whichever suppressor you choose to buy, make sure it has an Underwriters Laboratory Approval.

How Disks Store Information

Your computer is an electronic device and, as such, needs power to operate. The programs and information you are working with—a report you are typing with a word processor, or a spreadsheet, perhaps—reside in a temporary memory; when you turn off the computer, you lose the data that was there, unless you save it to a disk. Disks exist to let you store and retrieve information from one computer session to the next and to help you exchange information with other users. If you don't save information to disk before you turn off your computer, you will lose the information.

Disks are either nonremovable hard disks, which reside inside your computer's chassis, or removable floppy disks. Hard disks are much faster than floppies and have much greater storage capacity. Hard disks reside in a sealed container. Figure 1-6 illustrates what you would find if you could open your hard disk.

Floppy disks come in two sizes, 5 1/4 and 3 1/2 inches. Your floppy disk drive only supports one of the two disk types; you can't interchange them. Figure 1-7 illustrates the two floppy disk types and their drives.

The computer magnetically records information on your disk much like a tape recorder or VCR records songs or movies on tape. When you record a song on a cassette tape, you can play the song back immediately, a few days later, or even after months or years. Likewise, when you store information on a disk, you can access the information until you later erase it, or intentionally record other information over it. As a cassette tape only has space for a limited number of songs, your disk, too, has space restrictions.

If you look closely at a floppy disk, you will see portions of a circular plastic disk that looks like the same material as recording tape. The disk has a special coating onto which the computer can record information. Never touch the plastic disk. Doing so might damage the disk's surface and result in the loss of the information the disk contains. Figure 1-8 illustrates what you would find if you were to open up and look inside a 5 1/4-inch disk.

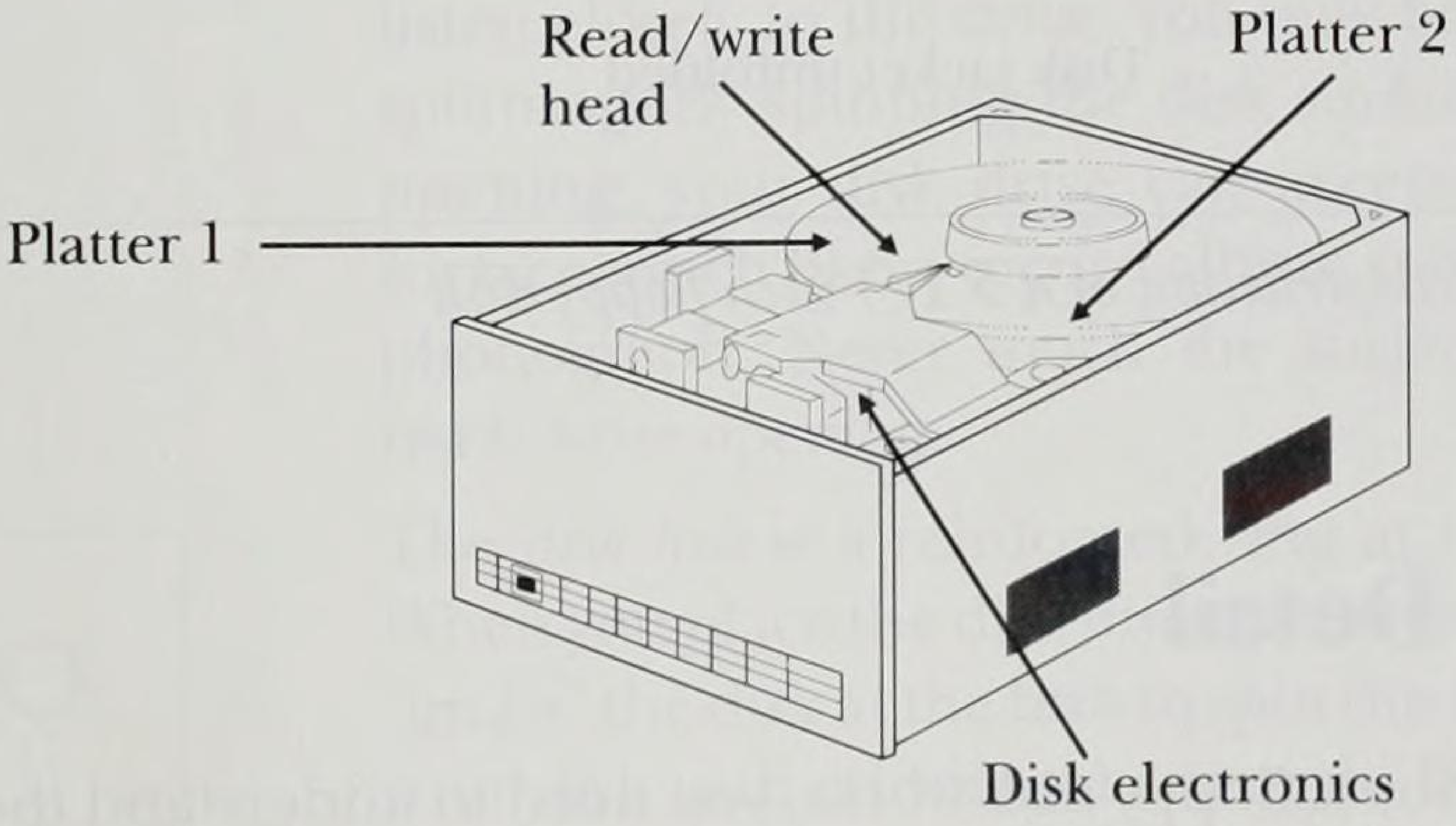


Figure 1-6. Cutaway drawing of a hard disk

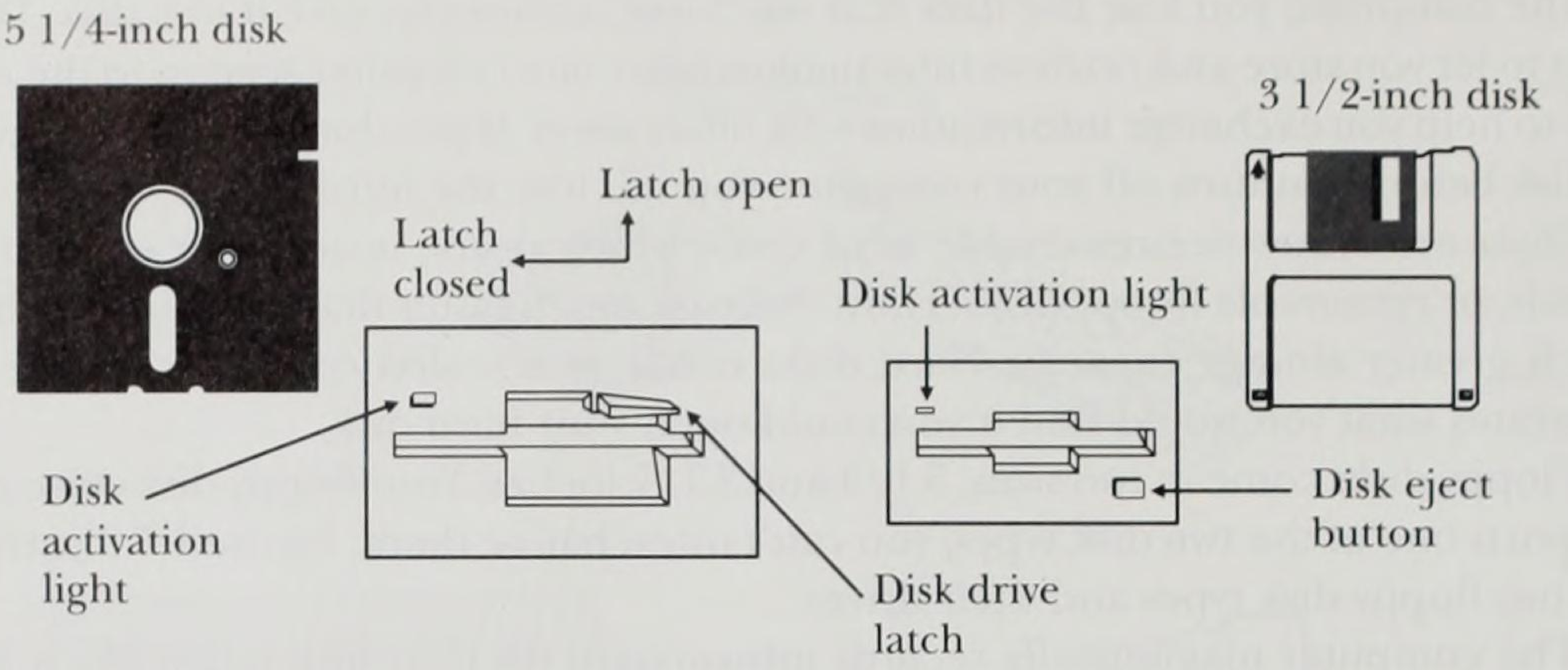

5 1/4-inch disk
3 1/2-inch disk
Latch open
Latch closed
Disk activation light
Disk activation light
Disk eject button
Disk drive latch

Figure 1-7. *5 1/4- and 3 1/2-inch floppy disks and their drives*

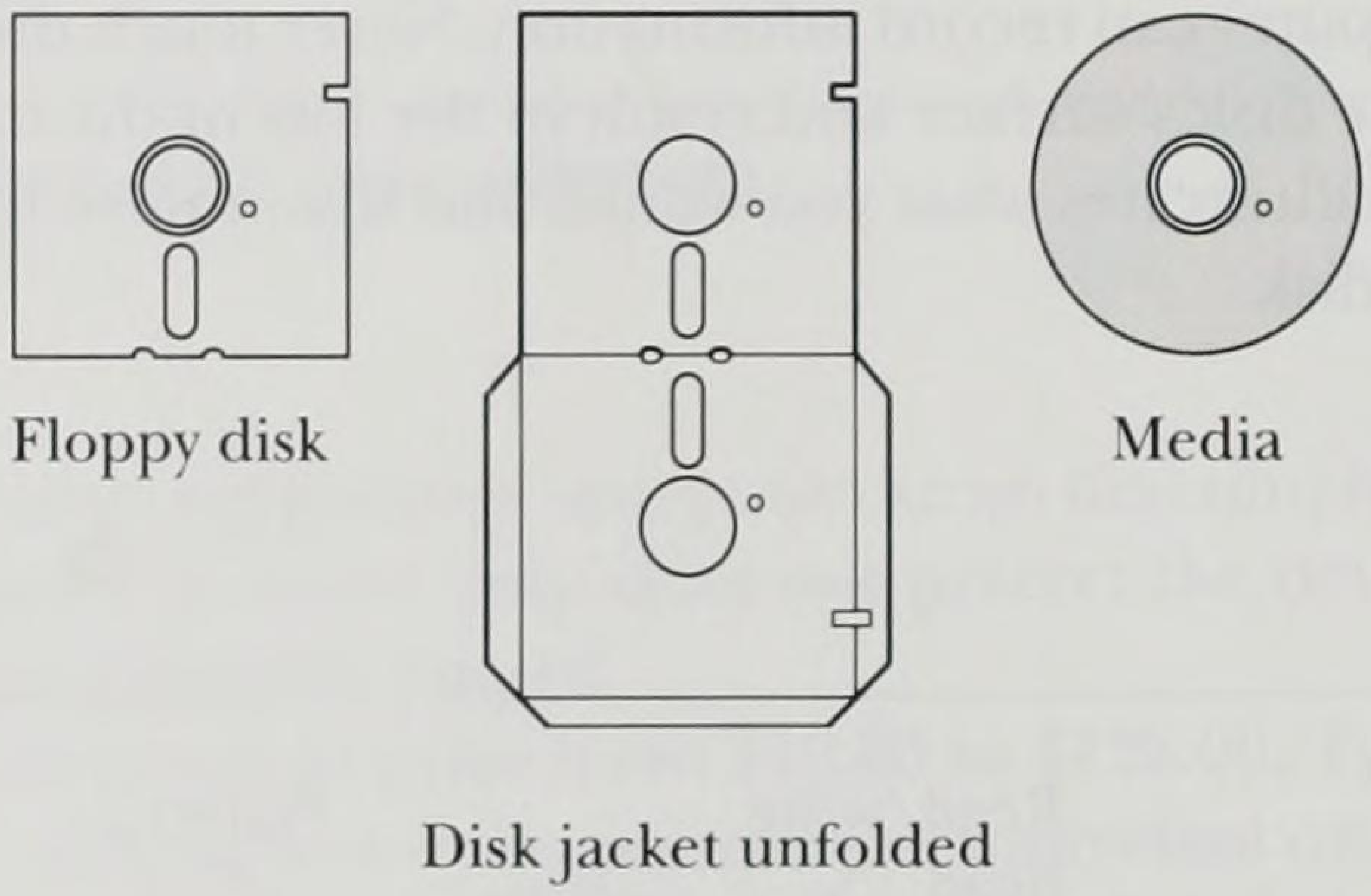
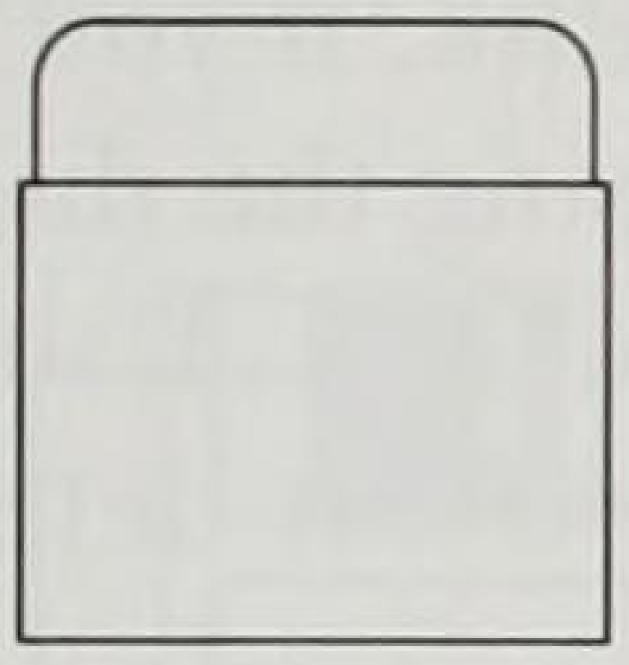

Floppy disk
Media
Disk jacket unfolded

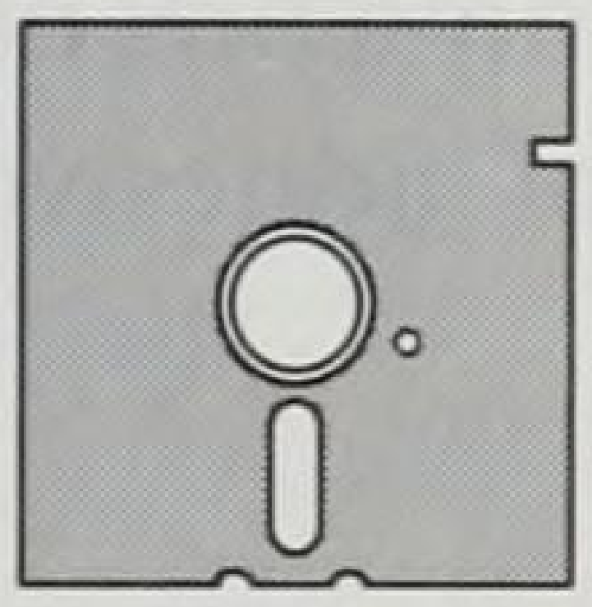
Figure 1-8. *The inner construction of a 5 1/4-inch floppy disk*

Floppy Disks in Detail

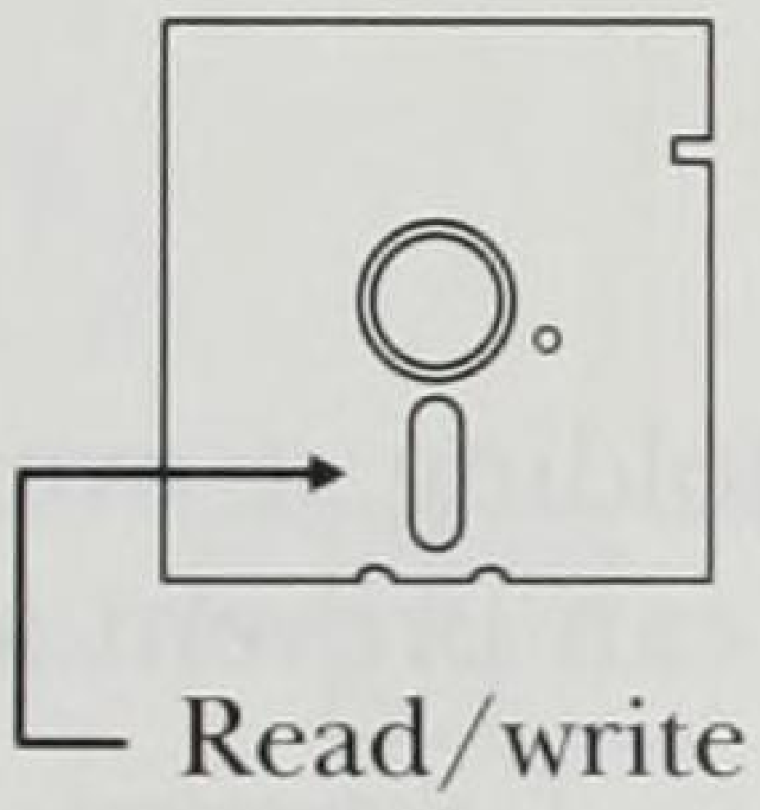
To better understand how a floppy disk works, you need to understand the disk's key components. Take out an unused 5 1/4-inch floppy disk (you don't want to endanger any data) and examine the disk as you read through the following discussion.



The *disk envelope* is a protective paper sleeve that you can slide your disk into when it is not in use. Because the sleeve is only paper, it cannot protect the disk from damage or possible bending resulting from other objects set on top of the disk. The envelope's purpose is to protect the disk from dust and minor spills. You need to remove the disk from its envelope before you insert the disk into the drive. Get in the habit of always placing a disk immediately back into its envelope. 3 1/2-inch floppy disks do not require envelopes, because they are surrounded instead by a protective hard-plastic cover.

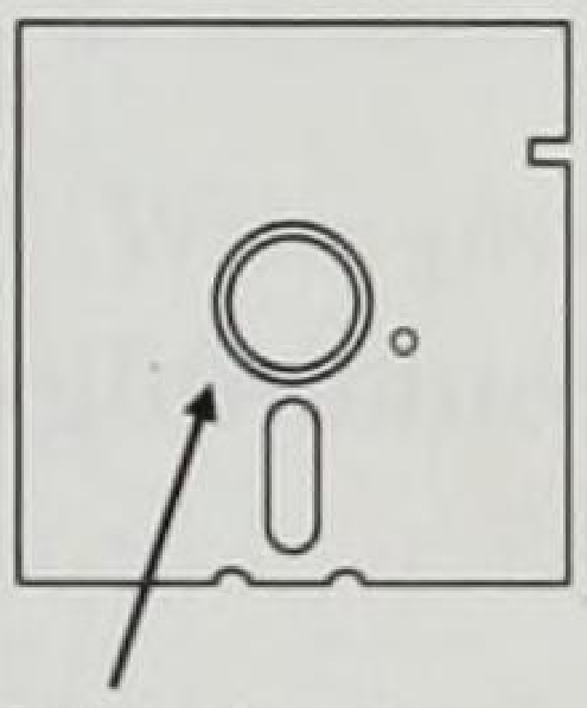


The *disk jacket* of a 5 1/2-inch floppy is the dark cardboard-like covering that surrounds the flexible plastic disk. The jacket serves two purposes. First, the jacket gives you a way to pick up the disk. As discussed, if you touch the disk's plastic surface, you risk damaging the surface and losing the information the disk contains. Second, the jacket gives the disk the stiffness you need to insert the disk into the disk drive. Be careful never to bend the disk by trying to force it into the drive.



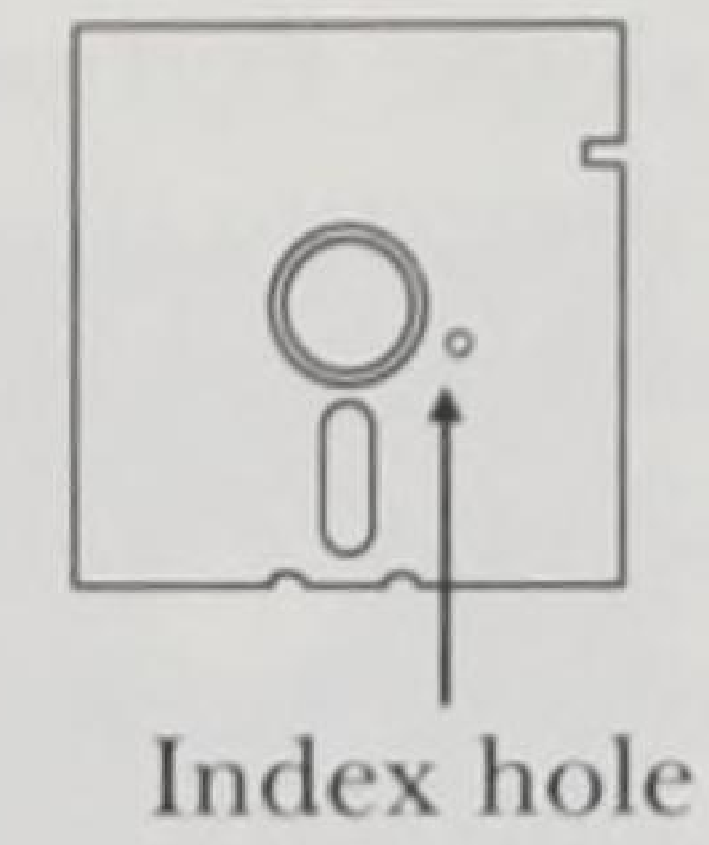
Read/write opening

The disk's *read/write opening* provides your computer's disk drive with access to the disk's recording surface, which is like the recording tape in a cassette—very thin and flexible, thus the name "floppy." *Writing* is the process of recording information on the disk. *Reading*, on the other hand, is the process of accessing the information the disk contains. When you insert a floppy disk into a drive, the drive spins the floppy disk at about 300 revolutions per minute. If you listen closely to the drive, you may be able to hear the disk spinning. By spinning the disk surface past the read/write opening, your disk drive can access the entire recording surface much like a record album spins past the needle of a phonograph. Never touch the surface of the disks at the read/write opening.

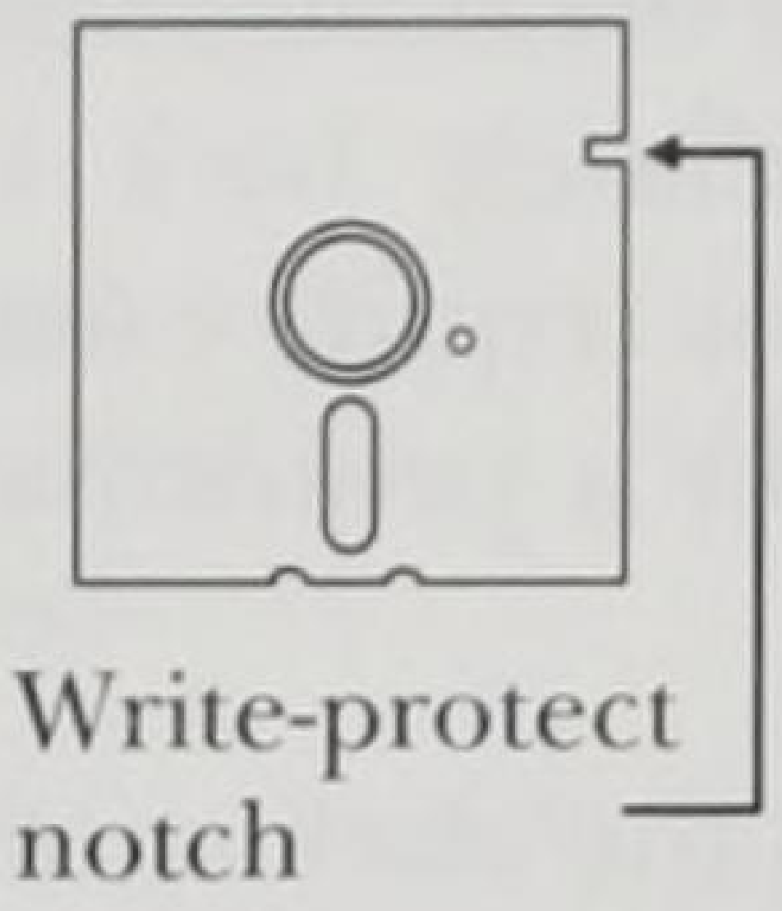


Disk hub

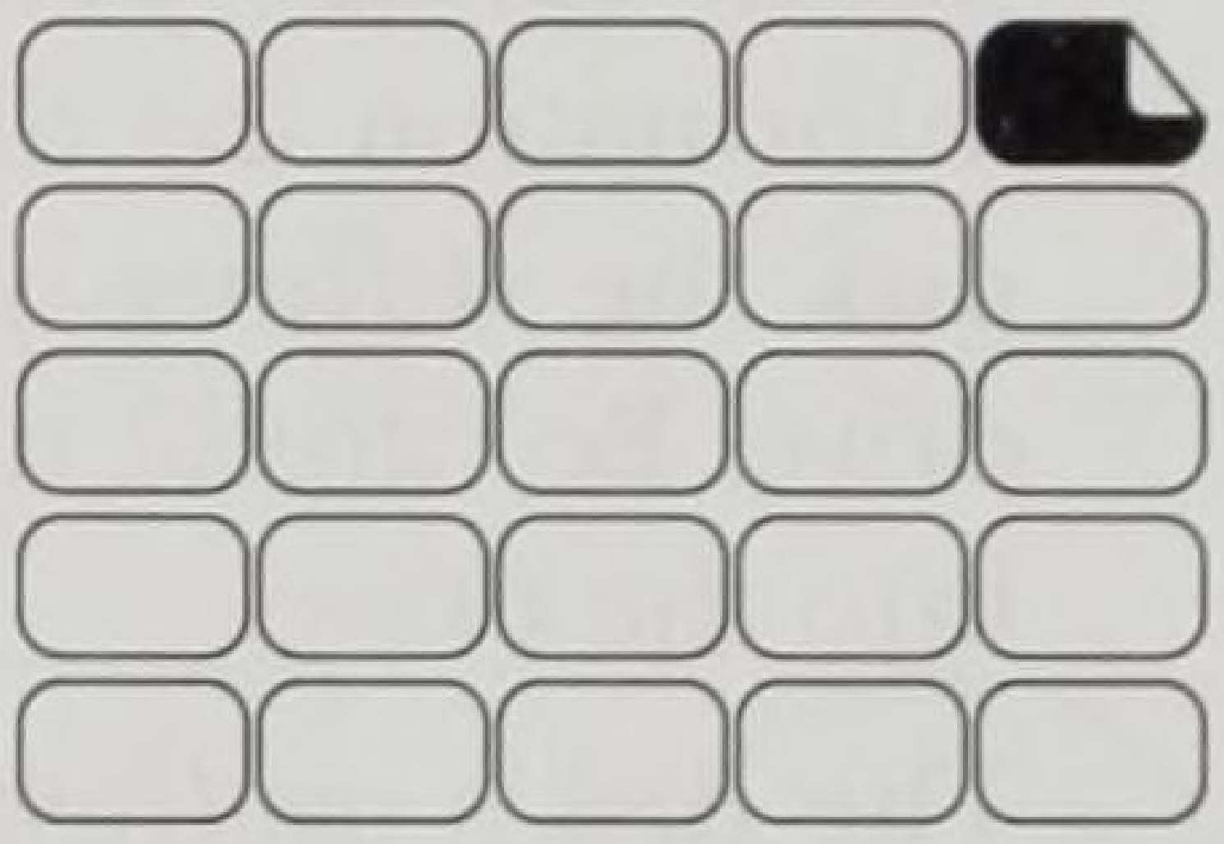
The *disk hub* is a reinforced ring at the center of the disk. When you place the disk into the drive, the disk drive actually "grabs" the disk at the hub to spin the disk. The hub provides your disk with additional strength. If your brand of floppy disk does not have a reinforced hub, you should consider purchasing a different brand of disk.



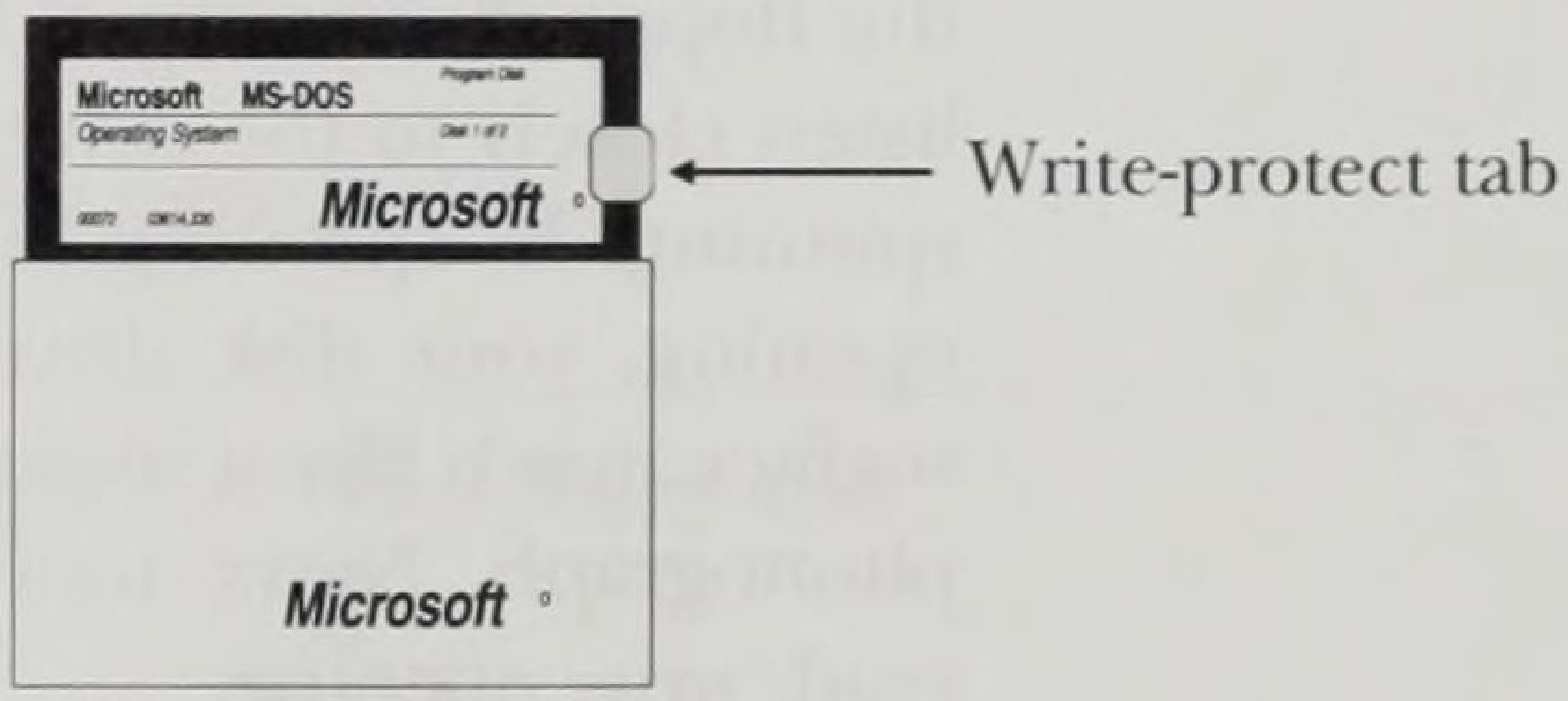
The *index hole* is a small opening that some disk drives use to determine their current position on the disk. If you gently spin your unused disk inside the jacket, using the disk hub, you will eventually find a small hole in the disk. Using this hole as a point of reference, the disk drive can determine the disk's current position.



The *write-protect notch* of a 5 1/2-inch floppy disk determines whether or not your disk drive can write or erase information to or from it. When the notch is exposed, the disk drive can read, write, or erase disk information. When the write-protect notch is covered, the disk drive can only read the information the disk contains. When you buy a box of floppy disks, the box should include a sheet of adhesive write-protect tabs like the one shown here:

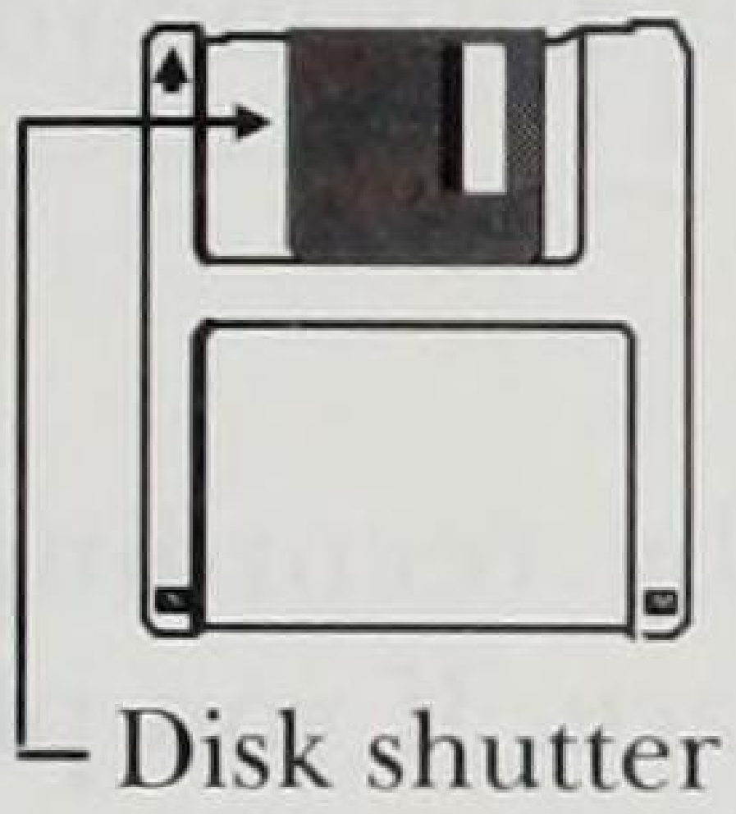


By peeling a tab off of the sheet and folding it over the write-protect notch, as shown here, you can prevent a disk from being written to or erased:

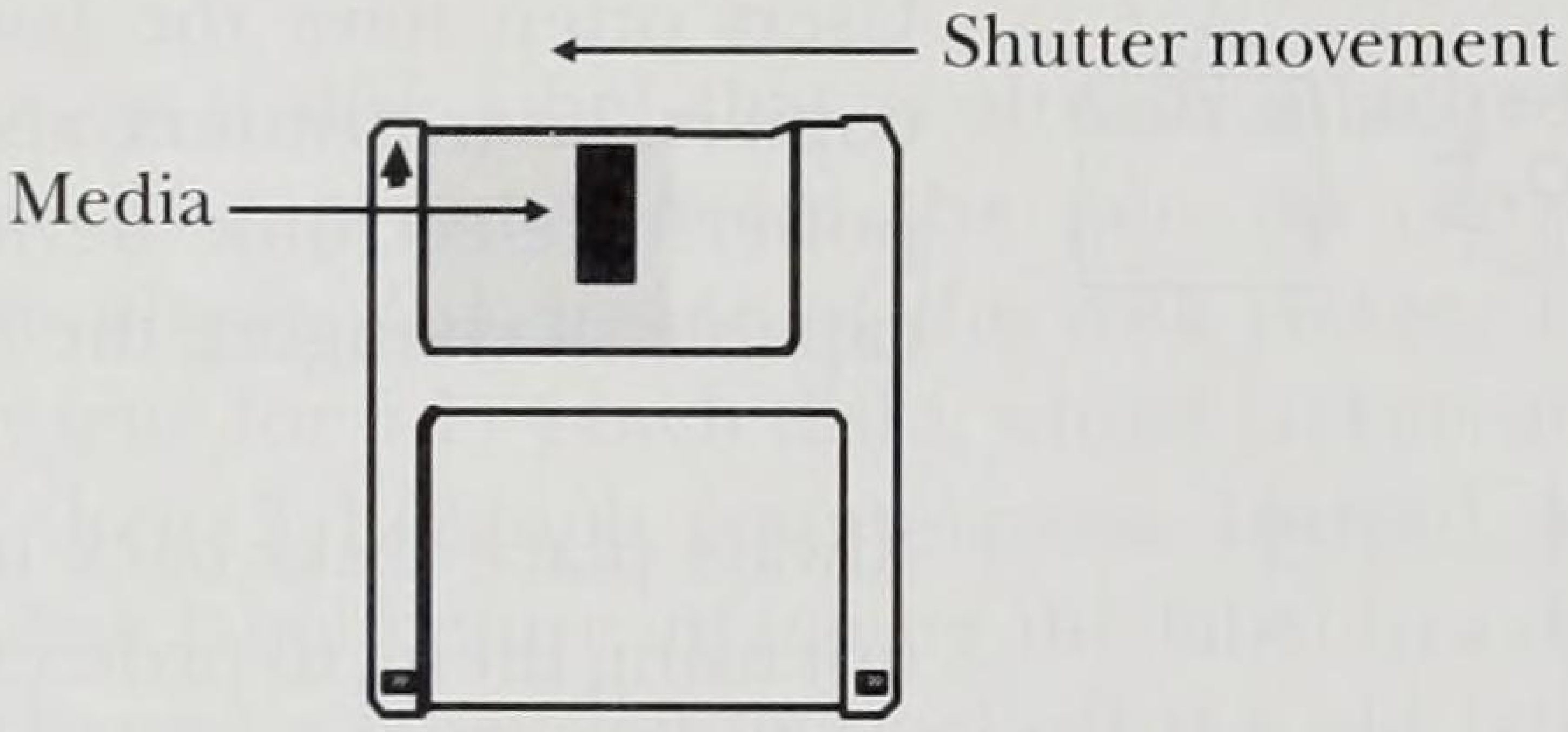


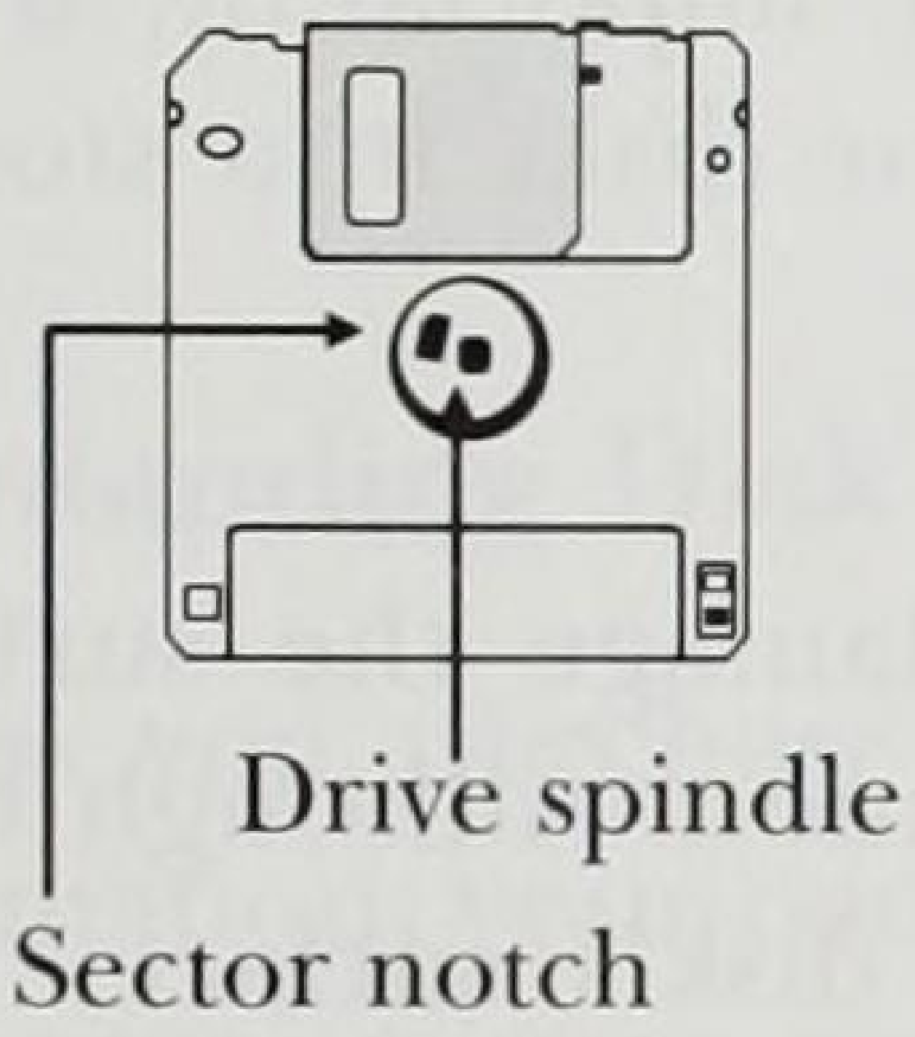
You should always write-protect the disks containing original copies of software that you buy.

A 3 1/2-inch floppy disk serves the same purpose as a 5 1/4-inch disk, but it is smaller, and its jacket is more rigid.

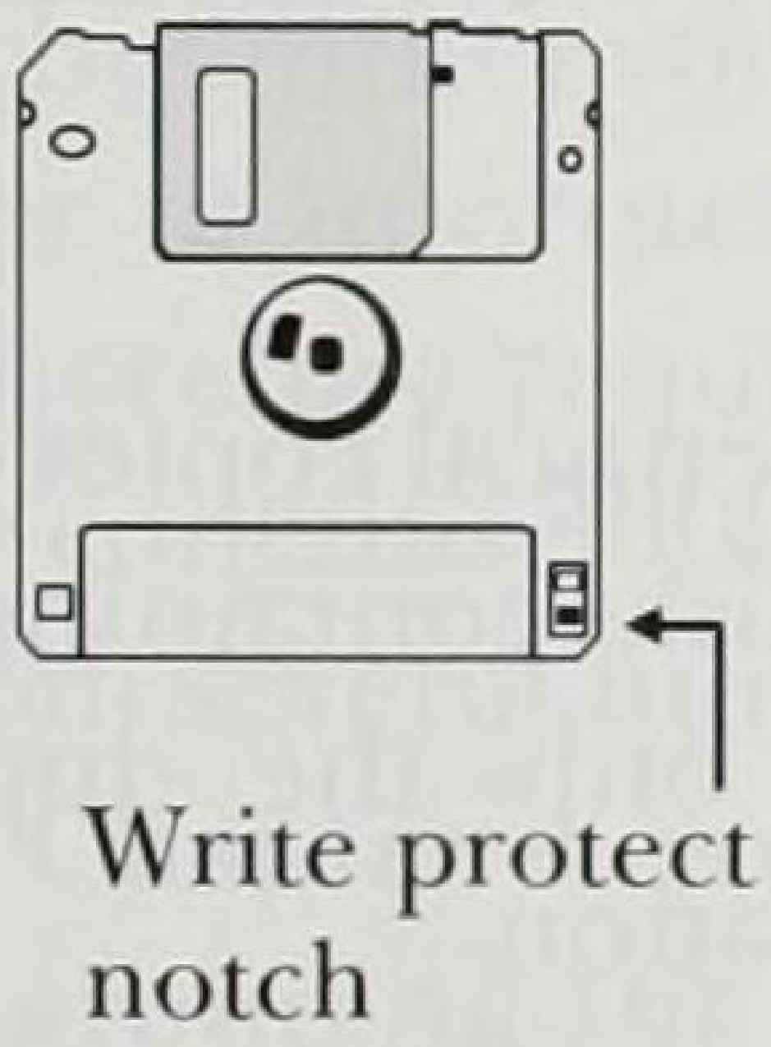


The 3 1/2-inch disk does not permanently expose the disk's surface at a read/write opening. Instead, the disk has a metal shutter that the disk drive slides open to access the disk's surface. Using an unused (don't endanger any current data) 3 1/2-inch floppy, hold it so that you are looking at the front of the disk and the shutter is pointing upward, then slide the shutter to the left to expose the disk's surface, as shown below. As before, do not touch the disk's surface. When you release the shutter, it springs closed.





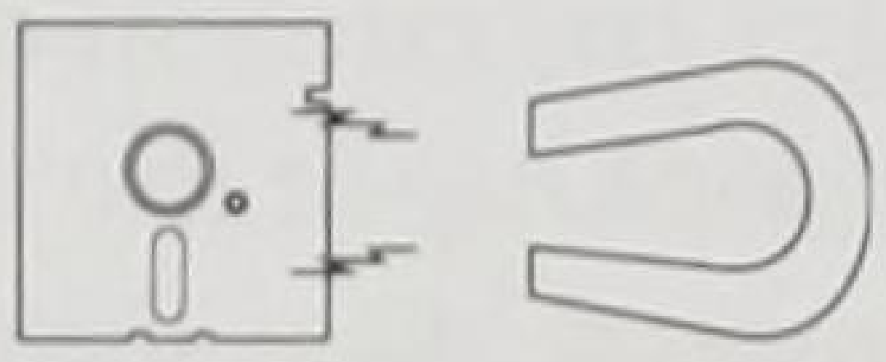
Unlike the 5 1/4-inch floppy disk drives, which grab the disk's hub to spin the disk, 3 1/2-inch disk drives use a spindle mechanism. The drive uses the small square hole to spin the disk, and the larger hole to determine its current disk position (much like the 5 1/4-inch disk's index hole).



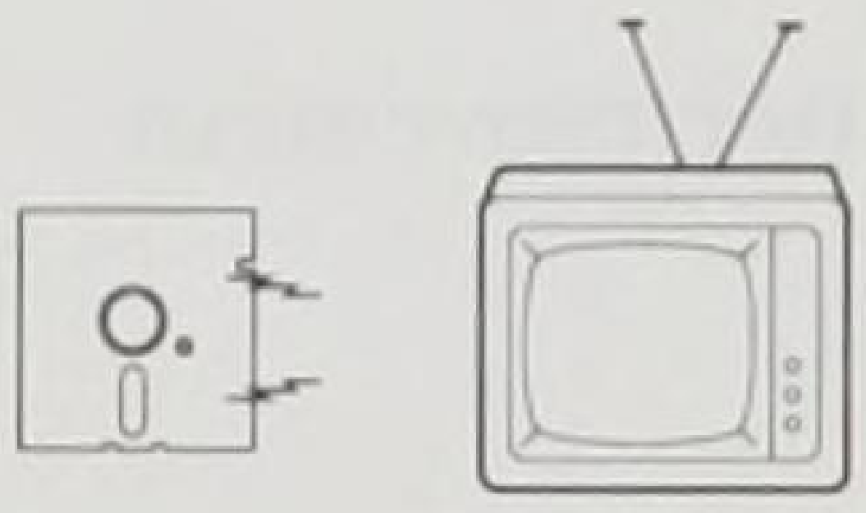
The write-protect notch on a 3 1/2-inch disk has its own built-in sliding cover. By sliding the cover back and forth, you can quickly write-protect or unprotect a disk. When you can look through the small hole in the disk, the disk is write-protected, and the disk drive can only read information from the disk. When the hole is covered, the disk drive can read, write, and erase information.

Caring for Your Disks

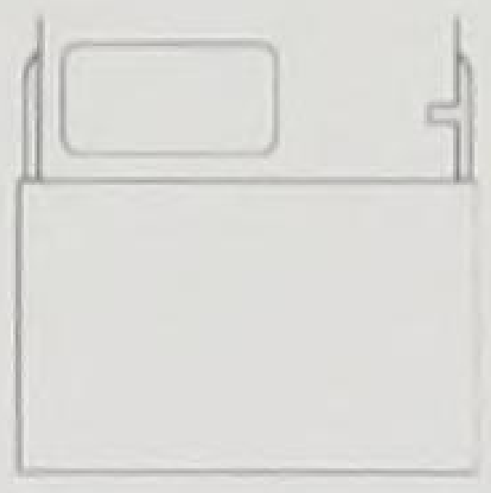
Although floppy disks are very durable, they can be damaged, resulting in the loss of information the disk contains. Follow these rules for handling your disks and you will greatly reduce the chance of damage.



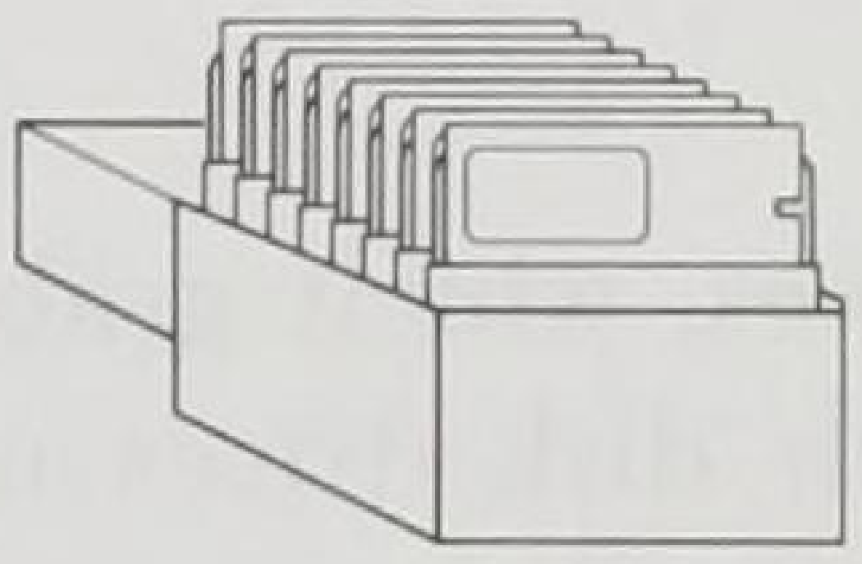
Your disk drive stores information by recording the information to the disk's magnetic surface. If you place a disk near a magnetic device, you risk inadvertently changing the disk's magnetized values. Many users ask if the X-ray machines at an airport can damage their disk. Airports claim no. Personally, I don't risk my key disks to such exposure.



Users often have the bad habit of setting disks down on copiers, laser printers, speakers, or even next to TVs. Such powerful electronic devices can produce a magnetic flux capable of changing the information on your disk.

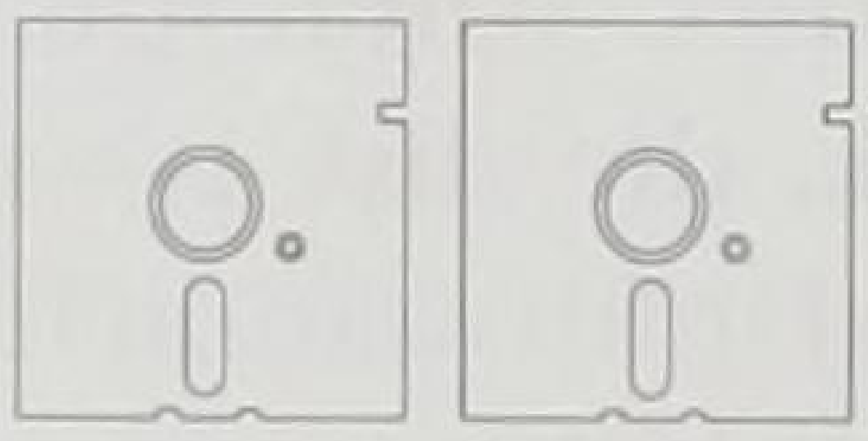


Always place disks back into a disk envelope when you are not using them, to protect them from dust, fingerprints, and minor spills.

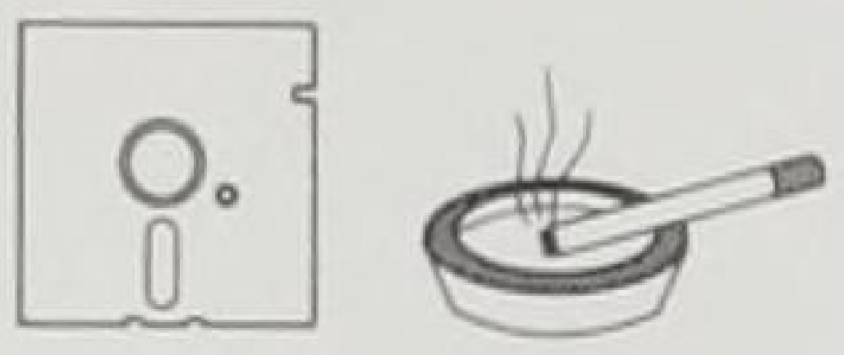


A disk storage container provides your disk with additional protection from dust and spills. More importantly, if you begin the habit of always placing your disks in a storage container, you won't misplace a disk.

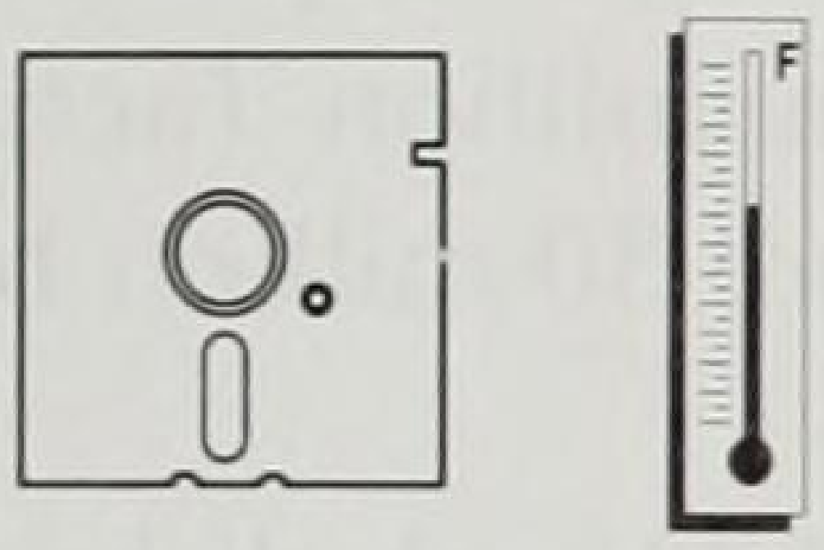
Touching your disk's surface can damage the surface, resulting in the loss of stored information. Although most users never intentionally touch the disk's surface, they inadvertently do so by picking up the disk near its read/write opening.



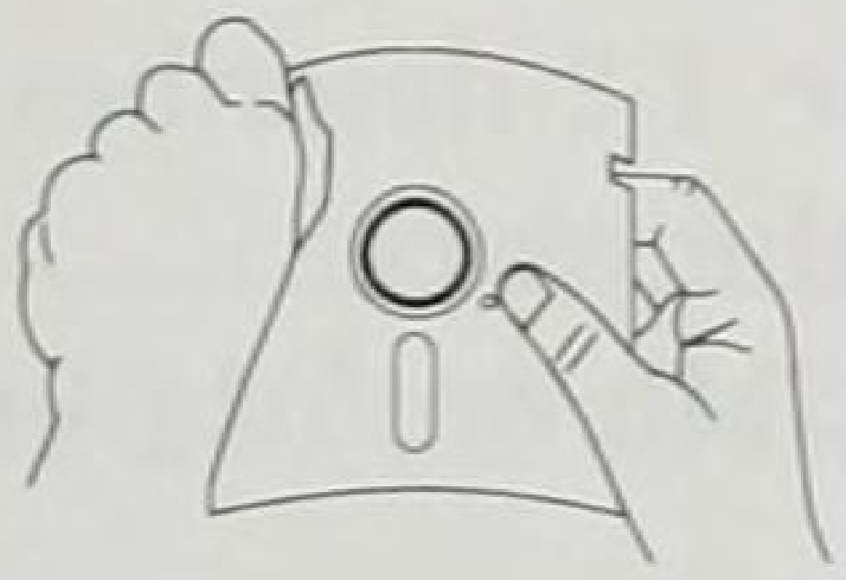
Chapter 12 teaches you how to make identical copies of your key disks. You should always duplicate disks containing newly purchased software programs, using only the duplicates while storing the originals in a safe location.



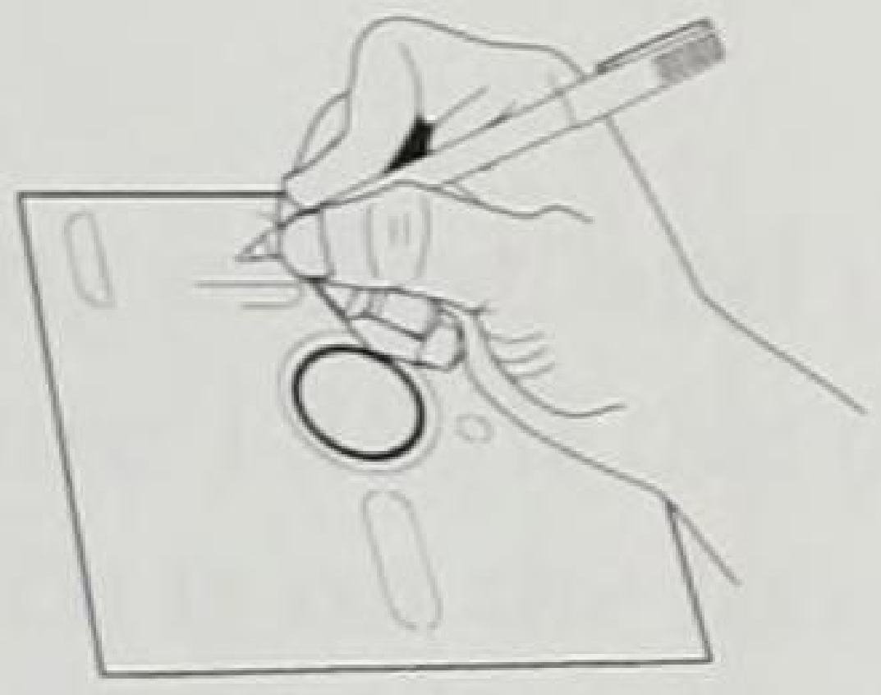
Smoke particles are one of the fastest ways to damage your disk's surface. The smoke particles not only collect on the disk, but inside the disk drive. Eventually these particles may scratch a disk, destroying the information the disk contains.



Exposing a disk to excessive cold or heat (direct sunlight) can cause the disk to expand or contract. Because the disk's surface contains millions of pieces of magnetized information, such changes in the disk's surface size can actually result in changes to the magnetized information. Many disk manufacturers recommend that the disks not be exposed to temperatures below 50 degrees or above 110 degrees Fahrenheit.



If you bend or crease a disk, the drive will no longer be able to spin the disk or correctly read the disk's surface. Users damage disks most often by forcing the disk into a drive.



Never write on a disk label that is already attached to a floppy disk. The pressure from the pen tip can easily damage the plastic disk inside of the disk jacket. This is particularly true for 5 1/4-inch disks, whose jackets are not as rigid as their 3 1/2-inch counterparts. Instead, always write your disk label before attaching the label to a disk. If you must change a label, gently peel off the old label, or simply attach the new label over the top of it. If you have a light touch and use a felt-tip pen, you can safely write on the disk label attached to the disk.

Understanding Disk Sizes

Just as you purchase cassette tapes that let you store 60, 90, or 120 minutes of information, your computer disk also has fixed storage capacities. When you discuss computer storage capabilities, however, you don't speak in terms of minutes of storage, but rather you use the term *bytes.* For our purposes, consider a byte as a single character of information. The word "disk" uses four characters, for example, and would therefore require four bytes of storage. A single-spaced, typed sheet of paper typically contains 4000 characters or bytes. Depending on your disk type, the disk can store from several hundred thousand to over one hundred million bytes of information. In Chapter 4, you will learn that DOS organizes the information you store on disk by grouping the information into files. All discussions about file sizes will be considered in terms of bytes.

To help you abbreviate your disk sizes, many users (and books) use the terms *kilobytes* and *megabytes.* The letter K represents the word "kilo" or 1000. Actually, 1K is 1024 bytes, but round numbers are easier to visualize. A 360K-byte disk, for example, can store approximately 360,000 bytes. It is common to see kilobytes expressed as simply *Kb.* In a similar way, the term "megabyte" stands for a million (actually, 1,048,576) bytes. A 30-megabyte hard disk, for example, can store 30 million bytes. The term "megabytes" often appears as simply *Mb.*

As discussed earlier, floppy disks come in two sizes. When 5 1/4-inch disks were first introduced for the PC in 1981, each disk could store 160Kb (about 160,000 bytes). Likewise, the first 3 1/2-inch disk could store 720Kb. As disk technology has improved over the years, the disk's storage capacity has also increased. The increased capacity has resulted from the capability of disk drives to access both sides of a disk, and to store information closer (more densely) together. Table 1-2 describes the various 5 1/4- and 3 1/2-inch disk sizes that have emerged over the years.

The most commonly used disk sizes today are 360Kb and 1.2Mb on 5 1/4-inch disks, and 720Kb and 1.44Mb on 3 1/2-inch disks.

Looking at Disk Drives

Before your computer can access the information a disk contains, the disk must be in one of your computer's disk drives. For nonremovable hard disks, the storage disks actually reside within the drive itself. For floppy disks, on the other hand, you must insert the disk into a drive of the appropriate size. Figure 1-9 illustrates a 5 1/4- and a 3 1/2-inch disk drive.

When you insert a 3 1/2-inch disk into a drive, insert the disk so that the disk spindle is facing down and the shutter end of the disk enters the disk first. To insert a 5 1/4-inch disk, insert the disk with the label facing up, the end with the disk window

Disk Type	Size	Storage Capacity
5 1/4	160 Kb	163,840 bytes
5 1/4	180 Kb	184,320 bytes
5 1/4	320 Kb	327,680 bytes
5 1/4	360 Kb	368,640 bytes
5 1/4	1.2 Mb	1,213,952 bytes
3 1/2	720 Kb	737,280 bytes
3 1/2	1.44 Mb	1,457,664 bytes
3 1/2	2.88 Mb	2,915,328 bytes

Table 1-2. Disk Storage Capacities

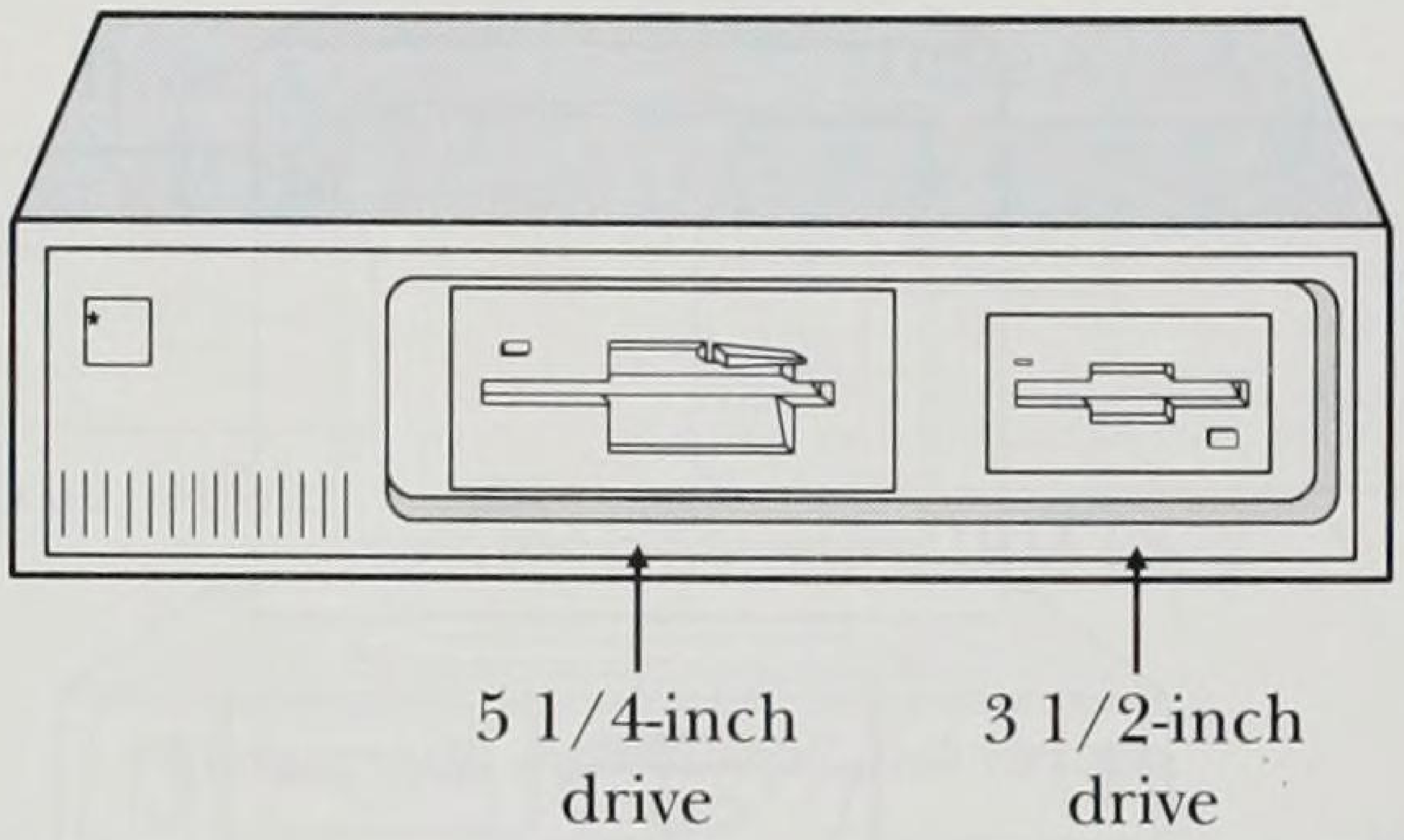
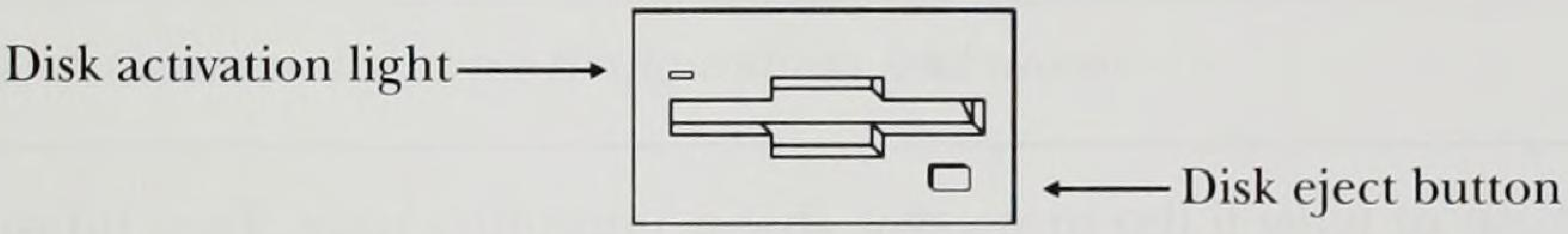


Figure 1-9. *5 1/4- and 3 1/2-inch drives*

going in first, and the end of the disk containing the label inserted last, as shown in Figure 1-10. If the disk does not have a label, insert the disk so that the write-protect notch is on the left side of the disk and at the end of the disk inserted last.

After the 5 1/4-inch disk is safely in the drive, close the disk drive latch (Figure 1-11). Latch types for 5 1/4-inch disks differ a bit; most common are the rotary latch shown in Figure 1-9 and the overhead latch shown in Figure 1-12.

To remove the disk from the drive, open the drive latch and gently remove the disk. If you are using a 3 1/2-inch drive, press the disk eject button, as shown here:



Note the small disk activation light on the front of your hard and floppy disk drives. Your computer illuminates this light whenever it is reading from or writing to a disk in the drive. Never turn off your computer or remove a disk while the disk activation light is on. If you do so, a critical disk operation might not complete, leaving your disk in an unusable state.

Disk Drives Have Single-Letter Names

Depending on your computer configuration, your computer will have one or more floppy disk drives and likely a hard disk. Each disk on your computer has a single-letter name. Your first floppy disk, for example, is drive A. Likewise, a hard disk is typically named drive C. If your computer has a second floppy drive, the drive is typically

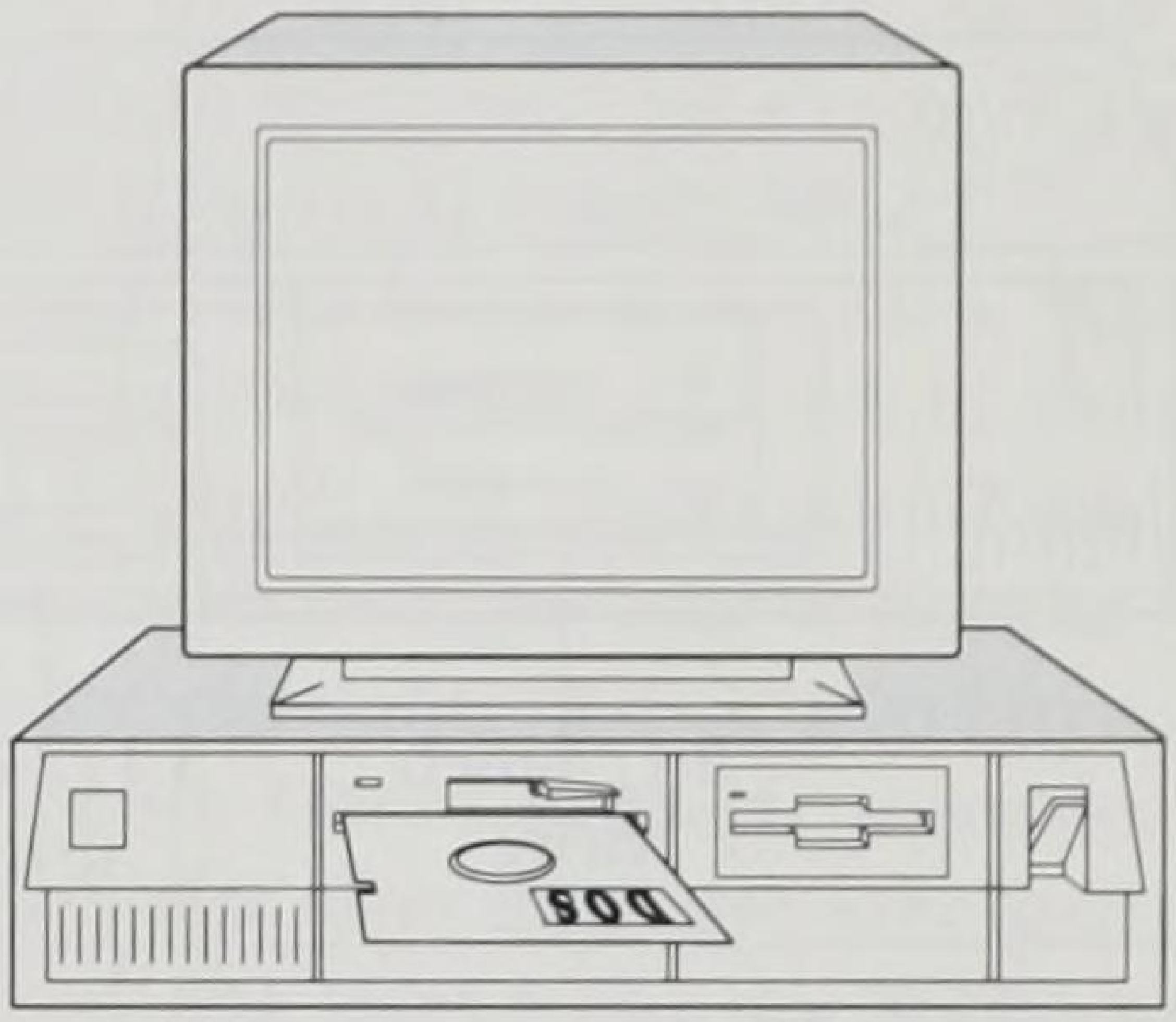


Figure 1-10. *Inserting a 5 1/4-inch floppy disk into the disk drive*

named drive B. A second hard disk might be named drive D. Figure 1-12 shows the name and location of common disk configurations.

Understanding Software

When most users think about the computer's processing power, they picture the computer's hardware. As it turns out, hardware only provides half of the story. To

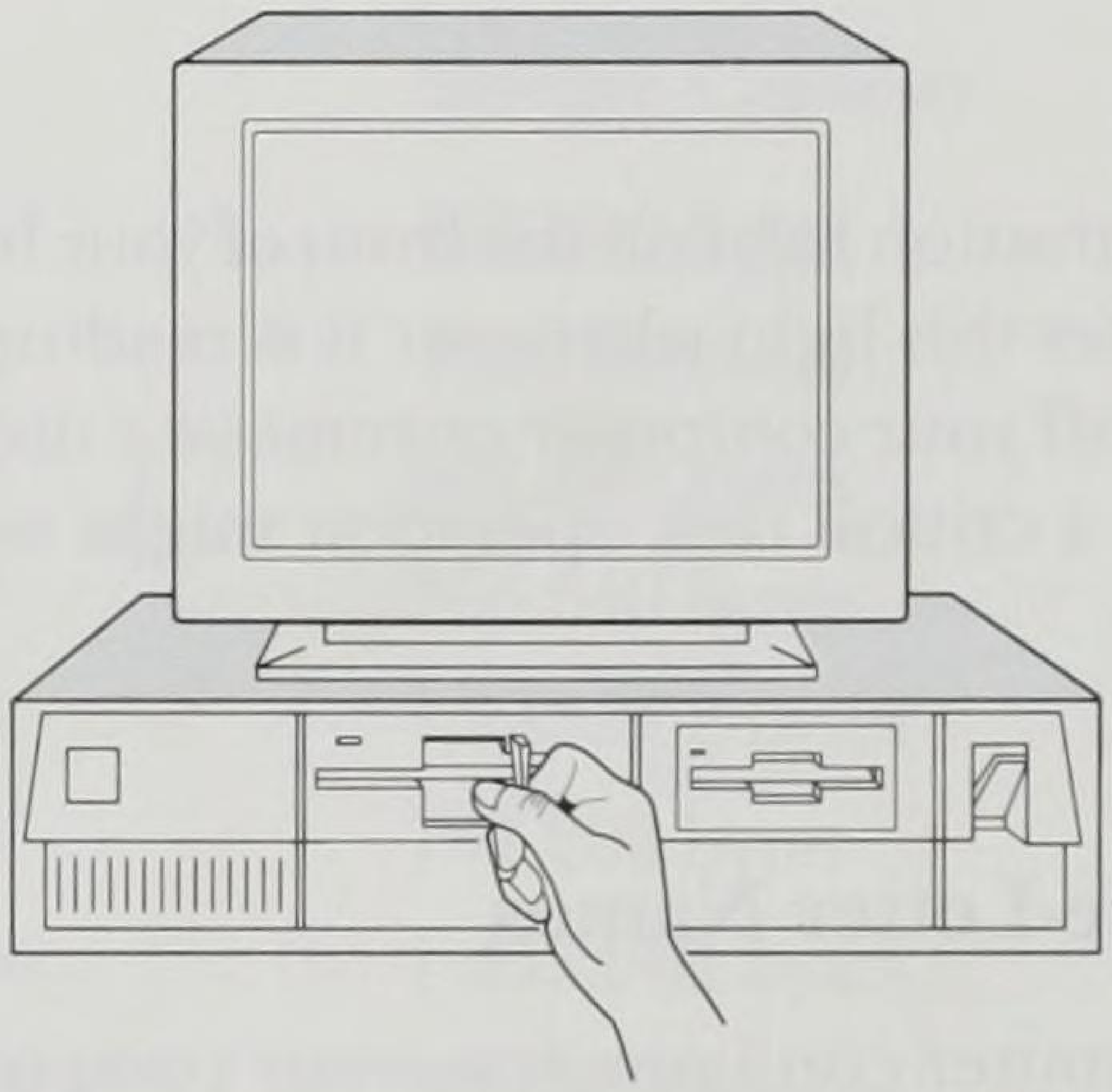


Figure 1-11. *After you place a 5 1/4-inch disk into the drive, you must close the drive latch*

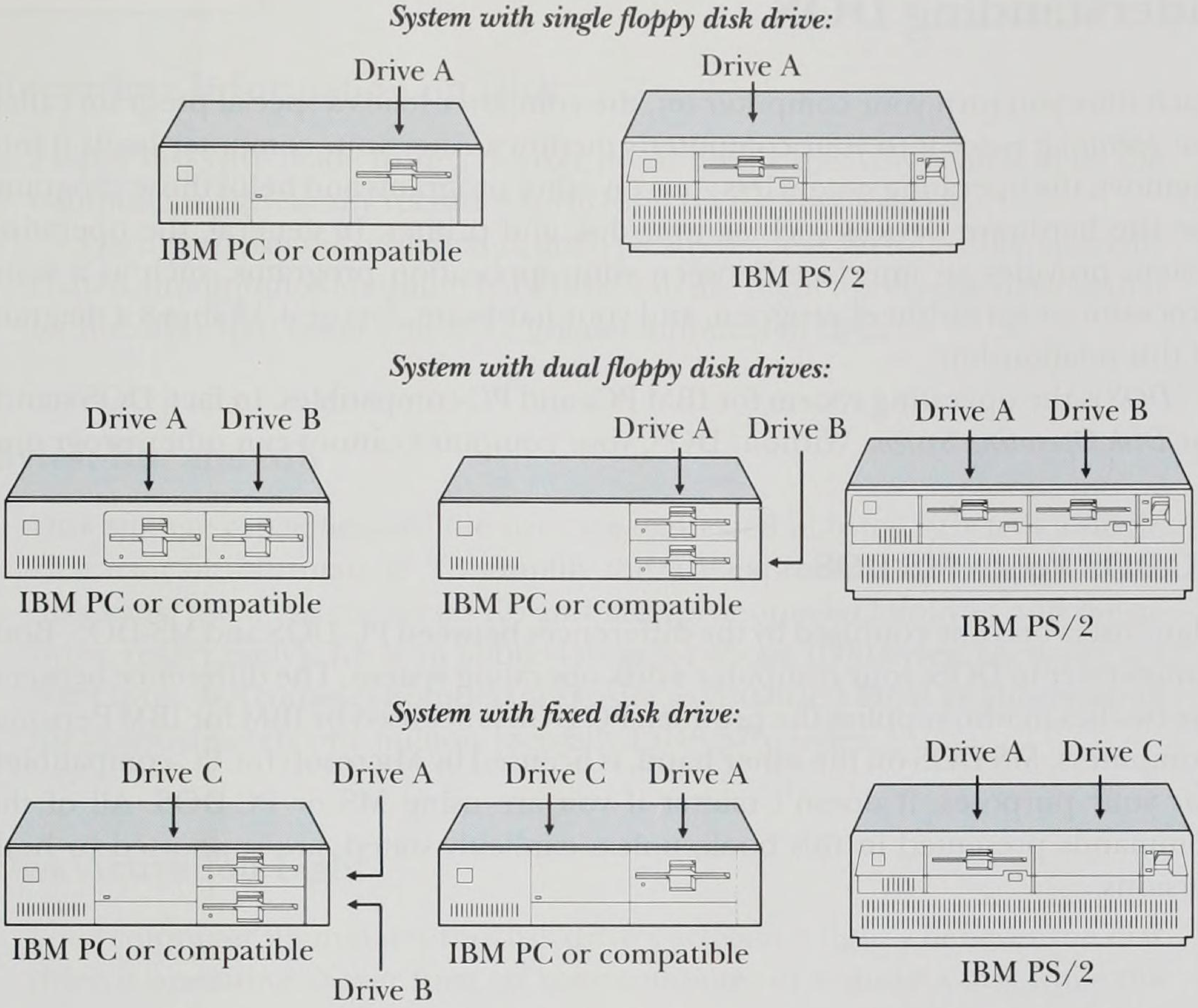

System with single floppy disk drive:
Drive A
IBM PC or compatible
Drive A
IBM PS/2
System with dual floppy disk drives:
Drive A
Drive B
IBM PC or compatible
Drive A
Drive B
IBM PC or compatible
Drive A
Drive B
IBM PS/2
System with fixed disk drive:
Drive C
Drive A
IBM PC or compatible
Drive B
Drive C
Drive A
IBM PC or compatible
Drive A
Drive C
IBM PS/2

Figure 1-12. *Common disk drive configurations and names*

perform useful work, your computer needs *software* to tell it what to do. Software is another name for computer programs. Popular computer programs include word processors, which let you type in reports and letters; spreadsheets, which let you organize budgets and other numerical data; database programs, which help you organize information; graphics and drawing programs; and even games!

A *computer program* is a list of instructions that tells your computer what to do. Without instructions, your computer's hardware will not do anything. A word processing program, therefore, is a list of computer instructions that lets you create, edit, and print documents. Computer software is stored on disks. When you buy a word processing program, for example, the program may require several disks. It's important to note that the disk *stores* software. The disk is not software. Just as a cassette tape stores music, the cassette tape is not music.

As you will learn in Chapter 3, you run a computer program by typing the program's name at the DOS prompt and pressing ENTER. DOS will then locate the program on your disk and load the program into your computer's fast electronic memory for execution.

Understanding DOS

Each time you turn your computer on, the computer loads a special program called the *operating system* into your computer's memory. After your computer loads it into memory, the operating system lets you run other programs and helps those programs use the hardware devices such as your disk and printer. In general, the operating system provides an interface between your application programs, such as a word processor or spreadsheet program, and your hardware. Figure 1-13 shows a diagram of this relationship.

DOS is the operating system for IBM PCs and PC-compatibles. In fact, DOS stands for *Disk Operating System.* Without DOS, your computer cannot run other programs.

PC-DOS Versus MS-DOS

Many users become confused by the differences between PC-DOS and MS-DOS. Both names refer to DOS, your computer's disk operating system. The difference between the two lies in who supplies the product. PC-DOS is licensed by IBM for IBM Personal Computers. MS-DOS on the other hand, is licensed by Microsoft for PC-compatibles. For your purposes, it doesn't matter if you are using MS or PC-DOS. All of the commands presented in this book, unless explicitly stated, are supported by both versions.

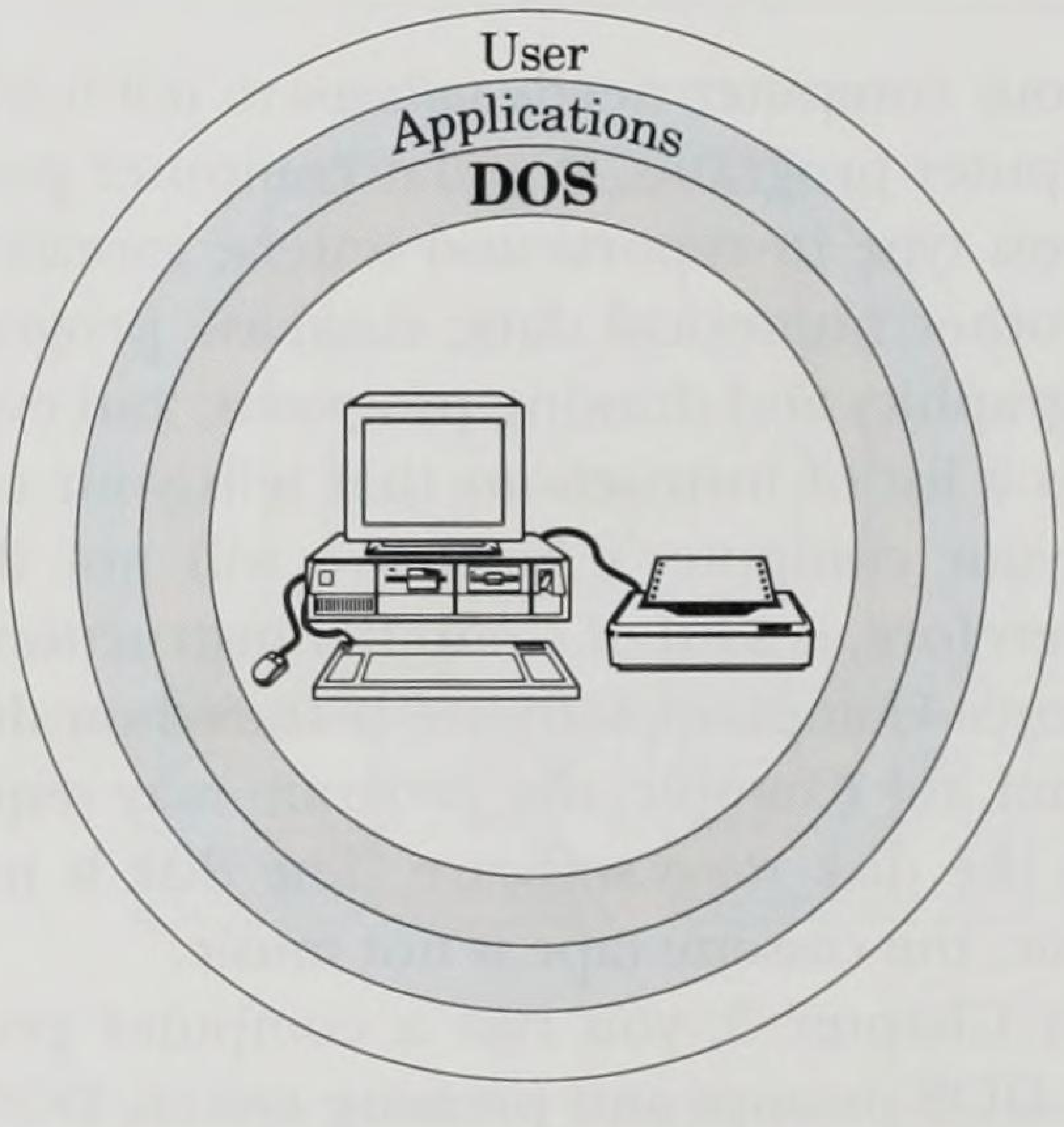


Figure 1-13. *The DOS operating system provides the hardware interface to your application programs*

SUMMARY

Recording Information on Disk

Floppy disks are made from a flexible mylar plastic specially coated to let the computer magnetically record information on the disk's surface.

Hard disks, on the other hand, are typically made from inflexible, specially coated aluminum. Aluminum hard disks can use higher-precision devices that let the disks spin faster and store greater amounts of data.

Bytes, Kb, and Mb

Disk storage capacities and file sizes are expressed in bytes. A byte is a single character of information. To simplify the size expression for larger file and disk sizes, users use the terms Kb and Mb (pronounced kilobytes and megabytes, respectively). Kb is an abbreviation for about 1000 bytes. (Actually, it's 1024 bytes, but rounder numbers are easier to visualize.) Mb is an abbreviation for approximately one million (actually 1,048,576) bytes.

Disk Activation Light

Your computer illuminates the disk drive's activation light whenever the disk drive is operating. Never turn off your computer or remove a disk while the activation light is on. Doing so may damage your disk, destroying the information the disk contains.

Software

Software is computer programs. A program is a list of instructions for your computer to perform. Common software programs include word processors, spreadsheets, and database programs. Computer software is stored on disk. The disks only store software; the disks themselves are not software.

What Is DOS?

DOS stands for Disk Operating System. Each time your computer starts, the computer loads DOS from disk into memory. After DOS is in memory, it lets you run other programs. DOS, then, is a special program that virtually every IBM PC and PC-compatible uses. Like all programs, DOS initially resides on a disk. The rest of this book takes a detailed look at DOS, its commands, and DOS techniques you can use to make your time at the computer more productive.

Hands On

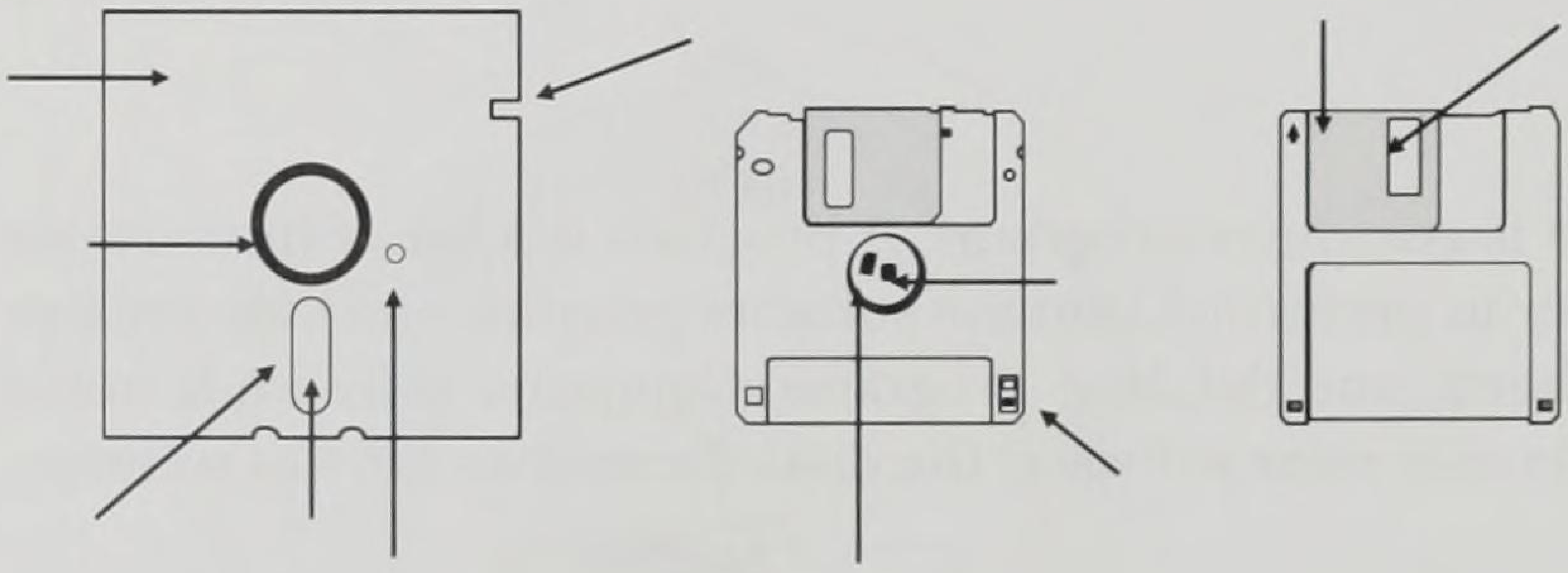
Using the disk drive figures presented in this chapter, determine your disk drive types (3 1/2 or 5 1/4) and whether or not you have a hard disk. In addition, determine your disk drive names (A, B, C, and so on). Use an unused floppy disk to experiment inserting and removing the disk into and out of the drive. If the disk is unlabeled, follow the guidelines specified in this chapter to ensure you are inserting the disk correctly.

Identify the disk's envelope, sleeve, jacket, index hole, write-protect notch, and disk hub. You might even consider sacrificing this disk by opening it up so you can examine the plastic disk

Last, check if your computer is plugged into a surge suppressor. If not, you should strongly consider buying one.

Review

1. What is hardware?
2. What is software?
3. What is DOS?
4. How does a hard disk differ from a floppy disk?
5. Label the parts of the following floppy disks.



6. Label the disk drives shown in Figure 1-14 with their correct names.
7. What is a program? List several common programs.
8. What is write-protecting a disk?

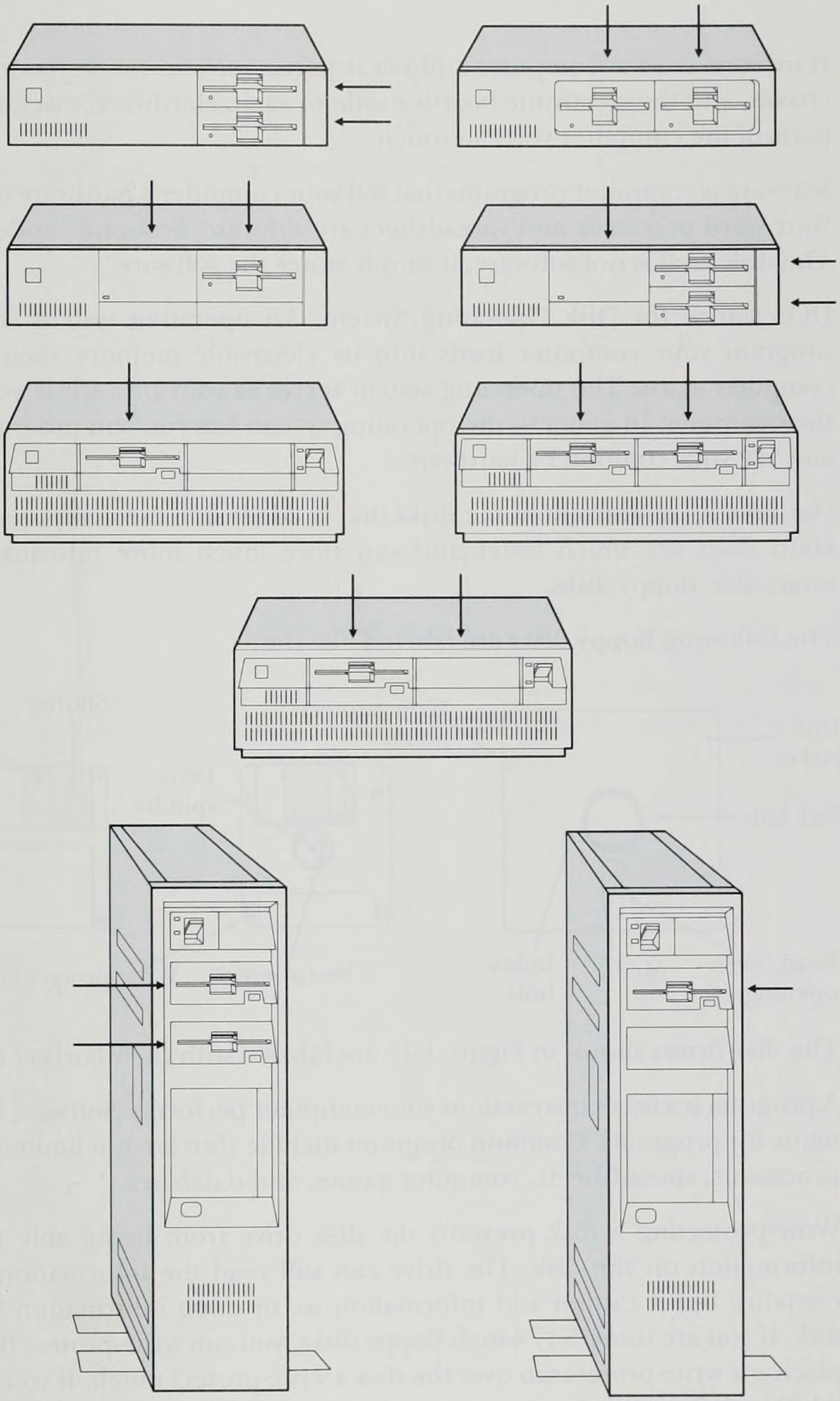
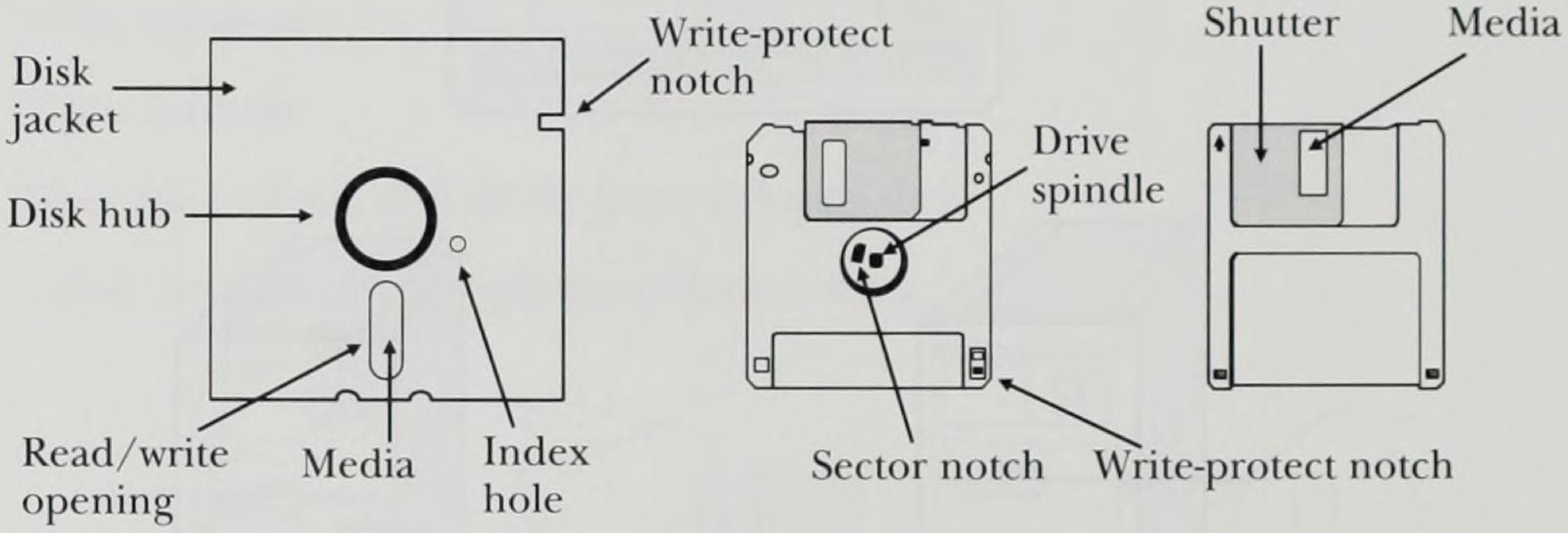


Figure 1-14. *Label the above disk drives with their correct names*

Answers

1. Hardware is your computer's physical parts, such as the keyboard, screen, chassis, and the electronic boards inside of each. Hardware, essentially, is the parts of the computer you can touch.
2. Software is computer programs that tell your computer's hardware what to do. Your word processor and spreadsheet are software. Software resides on disk. The disk itself is not software, it simply stores the software.
3. DOS stands for Disk Operating System. An operating system is a special program your computer loads into its electronic memory each time the computer starts. The operating system serves as your host while you work at the computer. In general, the operating system lets you run other programs, and use your computer's hardware.
4. Hard disks are nonremovable disks that reside inside the computer's chassis. Hard disks are much faster and can store much more information than removable floppy disks.
5. The following floppy disks are labeled like this:



6. The disk drives shown in Figure 1-15 are labeled with their correct names.
7. A program is a list of instructions your computer performs. Software is another name for programs. Common programs include (but are not limited to) word processors, spreadsheets, computer games, and databases.
8. Write-protecting a disk prevents the disk drive from being able to record information on the disk. The drive can still read the information the disk contains, but it cannot add information to, or erase information from, the disk. If you are using 5 1/4-inch floppy disks, you can write-protect the disk by placing a write-protect tab over the disk's write-protect notch. If you are using 3 1/2-inch floppy disks, you can write-protect a disk by sliding up the write-protect cover to expose a hole through the disk cover.

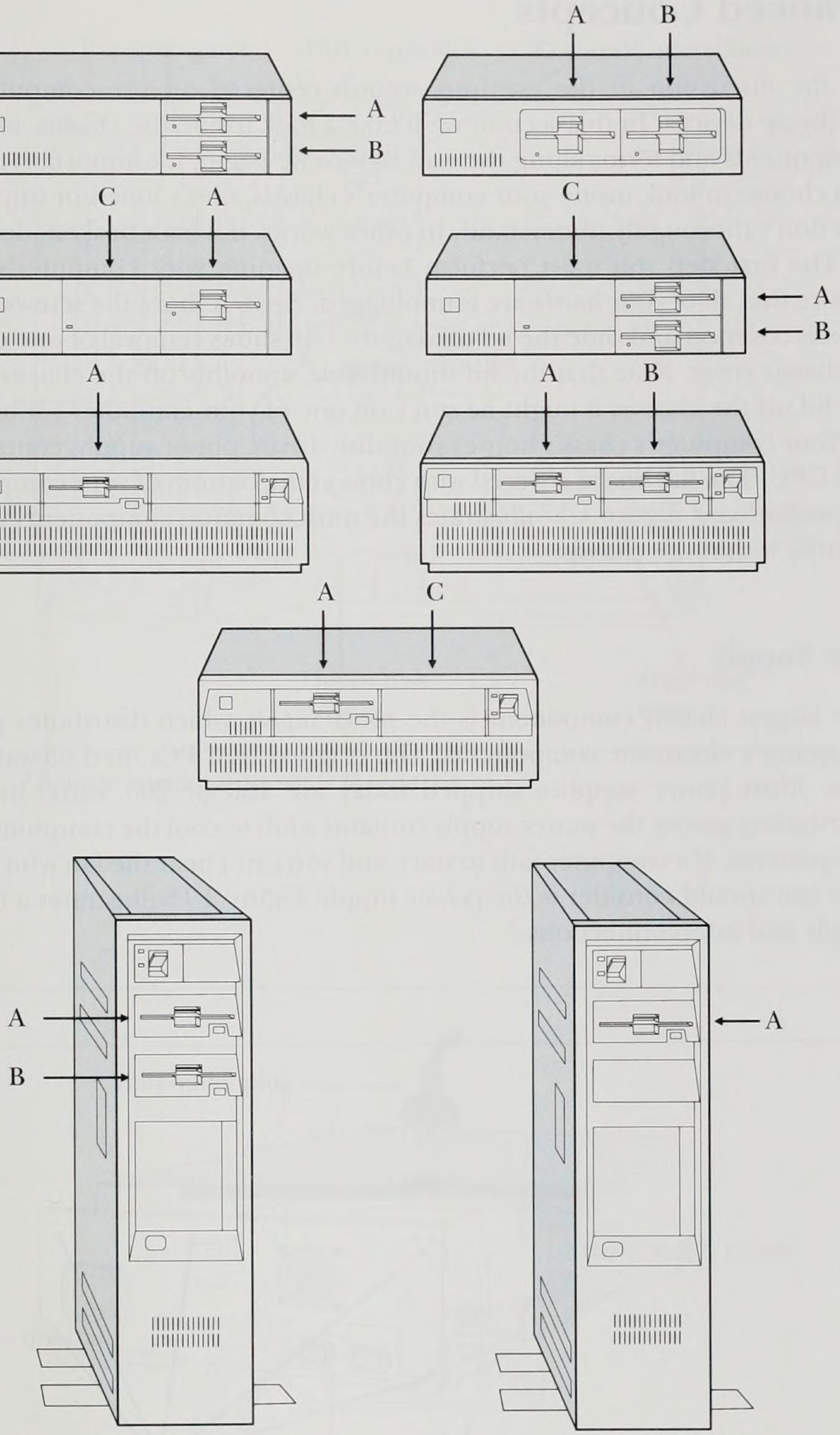

A
B
A
B
C
A
C
A
B
A
A
B
A
C
A
B
A

Figure 1-15. *Answer key for Review question 6*

Advanced Concepts

All the discussion in the previous section centered on the computer's external hardware devices. In this section we'll take a look inside the chassis, identifying key components and terms along the way. Before we begin, it's important to note that if you choose to look inside your computer's chassis, don't touch or unplug anything you don't thoroughly understand. In other words, if it isn't broken, don't fix it.

The first step you must perform before opening your computer's chassis is to ensure that all of your hardware is unplugged. Next, remove the screws that hold the chassis cover on and slide the lid off. Figure 1-16 shows removal of one common type of chassis cover. Note that the lid should slide smoothly off the chassis. Never force the lid off the chassis; it might be stuck on one of your computer's internal cables.

Your computer's chassis houses your disk drives, power supply, controller boards, and CPU. The flat board covered with chips at the bottom of your computer is called the *motherboard*. Figure 1-17 illustrates the most common components you can easily identify within the chassis.

Power Supply

The largest chassis component is the *power supply*, which distributes power to the computer's electronic components. The original IBM PCs used 65-watt power supplies. Most power supplies shipped today are 150 or 200 watts. In addition to distributing power, the power supply contains a fan to cool the computer's electronic components. If a computer fails to start, and you can't hear the fan whirring, the first item you should consider is the power supply. Figure 1-18 illustrates a typical power supply and interconnections.

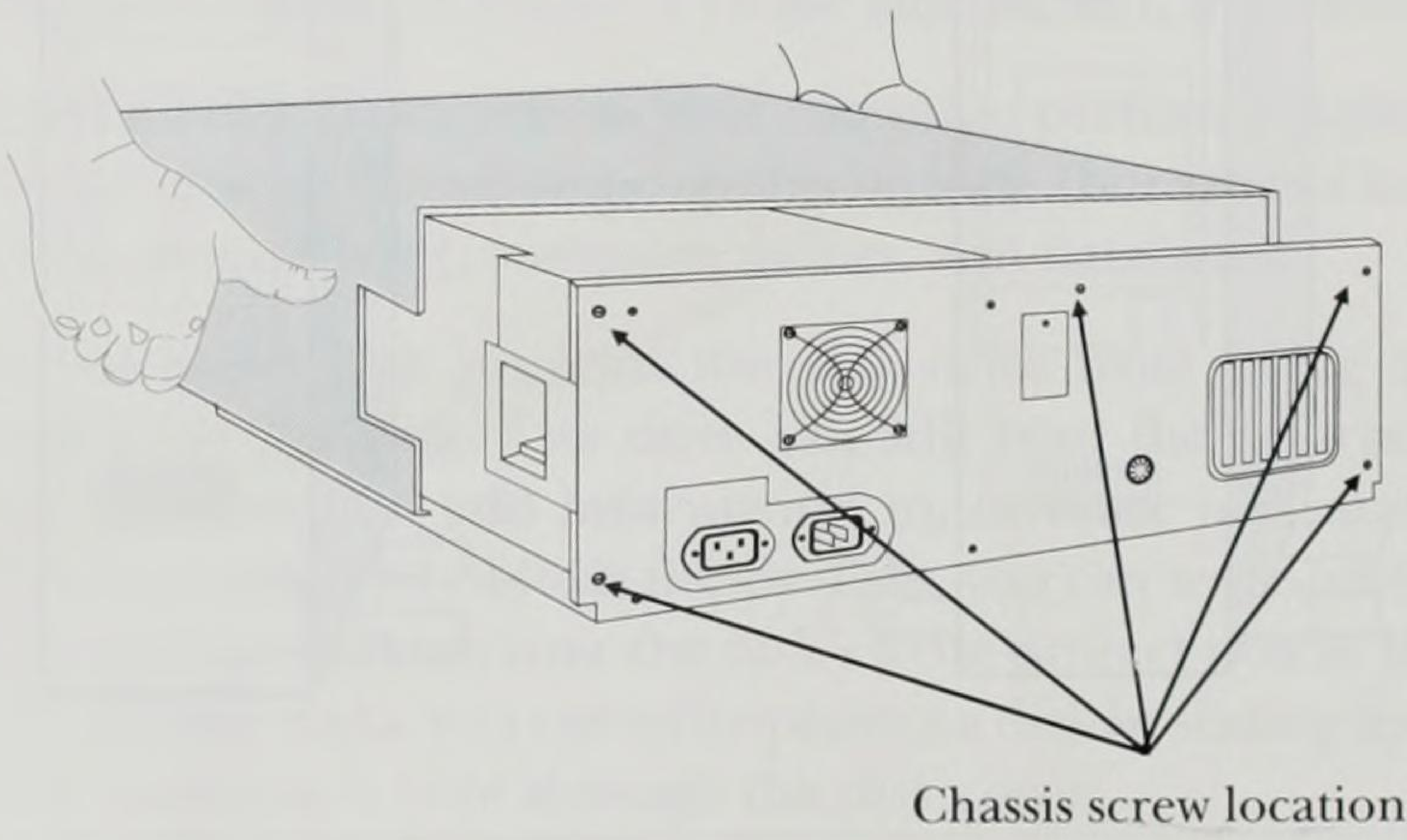


Figure 1-16. *After you remove the screws*

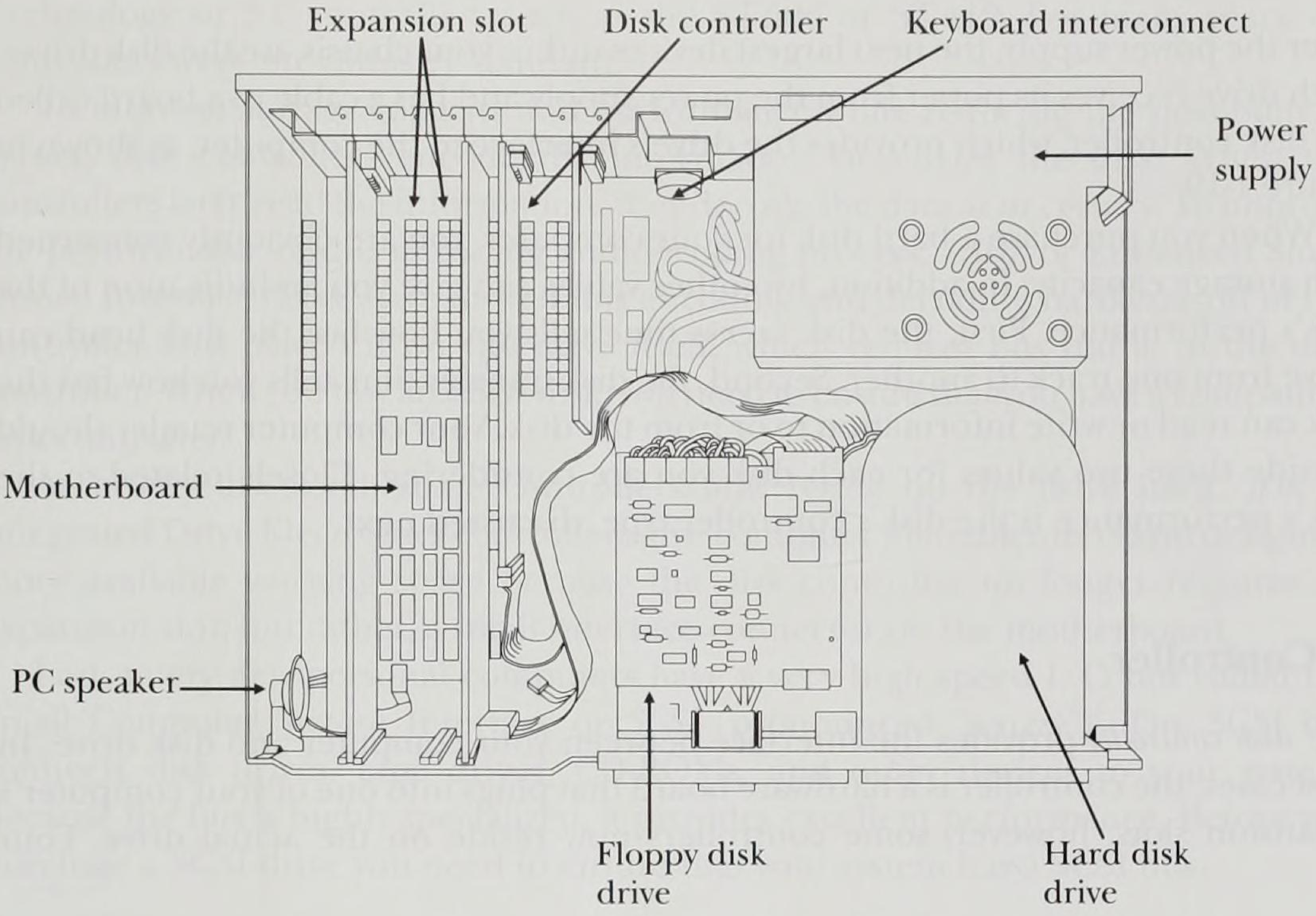

Expansion slot
Disk controller
Keyboard interconnect
Power supply
Motherboard
PC speaker
Floppy disk drive
Hard disk drive

Figure 1-17. *Common components inside the chassis*

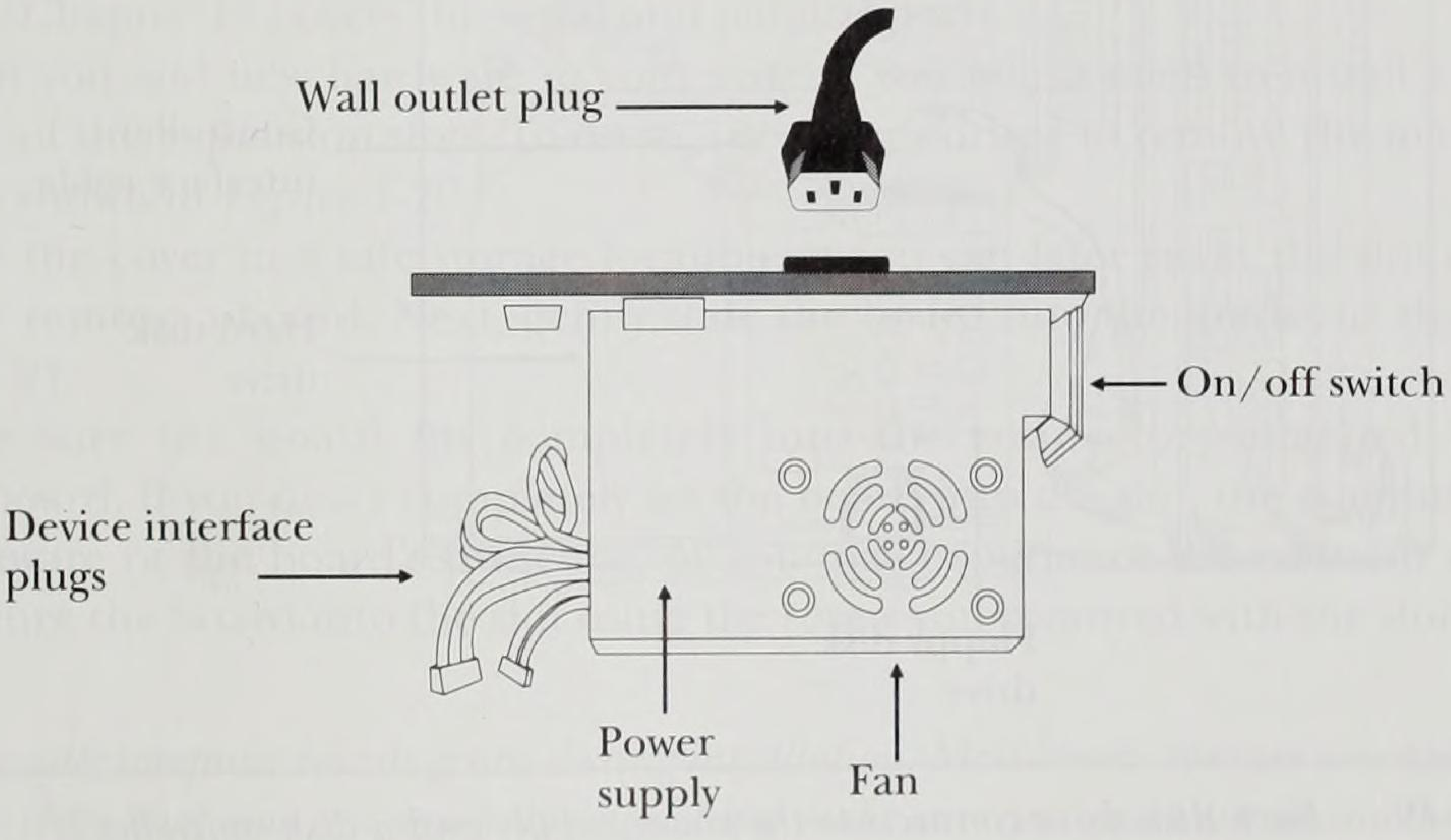

Wall outlet plug
On/off switch
Device interface plugs
Power supply
Fan

Figure 1-18. *Typical PC power supply*

Disk Drives

After the power supply, the next largest devices within your chassis are the disk drives. Each drive receives its power from the power supply and has a cable to a board called the disk controller, which provides the drive's interface to the computer, as shown in Figure 1-19.

When you purchase a hard disk for your computer, you are obviously concerned with storage capacity. In addition, two other values can give you an indication of the disk's performance. First, the disk access time tells you how fast the disk head can move from one track to another. Second, the disk transfer time tells you how fast the disk can read or write information to or from the disk. Your computer retailer should provide these two values for each disk you are considering. Closely related to the disk's performance is the disk's controller type, discussed next.

Disk Controller

The *disk controller* provides the interface between your computer and disk drive. In most cases, the controller is a hardware board that plugs into one of your computer's expansion slots; however, some controllers now reside on the actual drive. Four

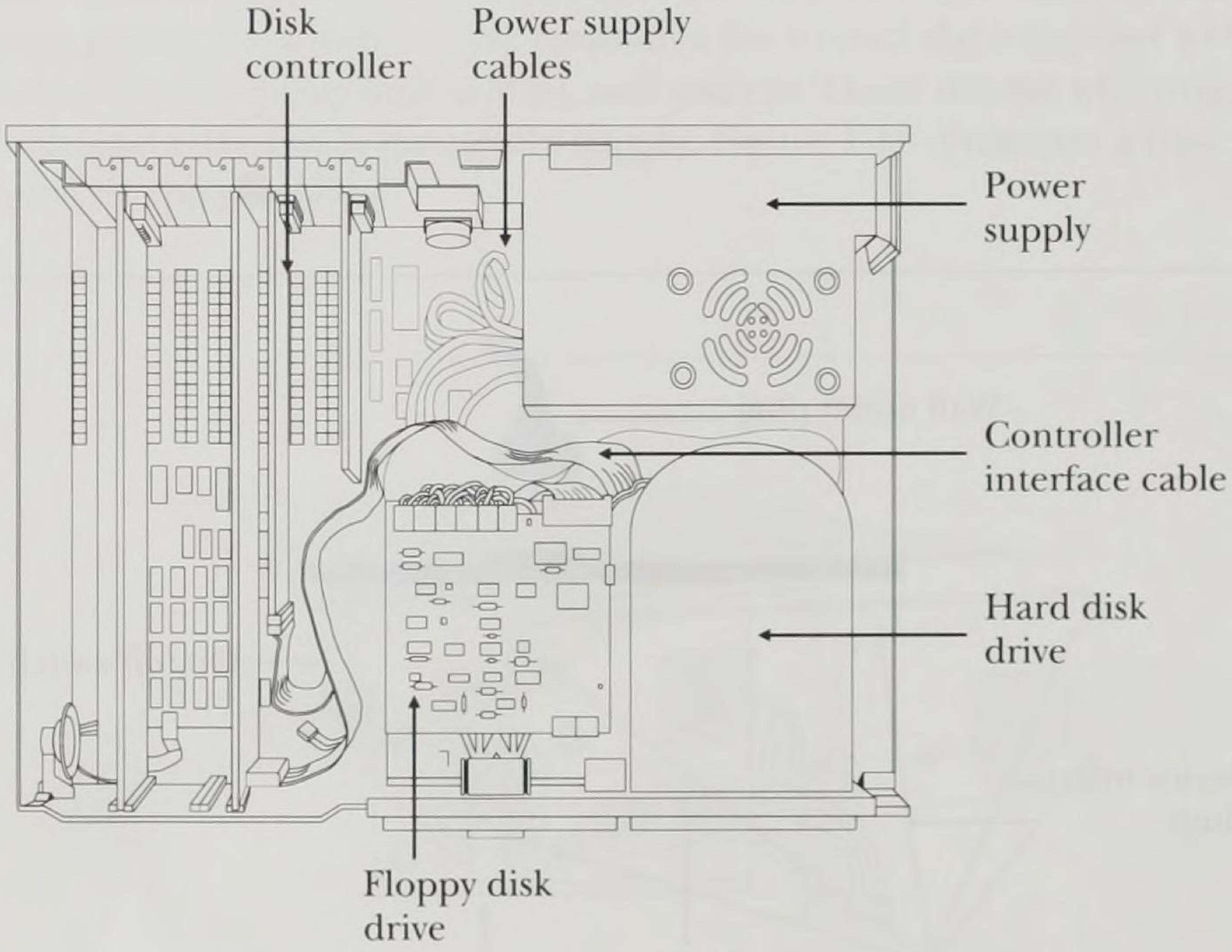


Figure 1-19. *Each disk drive connects to the power supply and a disk controller*

primary controller types dominate the industry. First, older systems use Shugart Technology or ST controllers such as the ST-506 or ST-412. For many years, ST controllers were the industry standard.

To increase storage capacity and performance while reducing the possibility of errors, disk controllers encode the information stored on the disk. When the controllers later read the information, they decode the data as necessary. To improve the performance of this encoding and decoding process, *ESDI*, or Enhanced Small Device Interface disks, have moved this encoding and decoding hardware off of the controller and placed it on the drive itself, which reduces bus traffic to the disk controller. When you buy an ESDI disk, you need to ensure that you have a compatible disk controller.

As briefly discussed, some controllers now reside on the drive itself. *IDE* or Integrated Drive Electronics controllers give computer manufacturers and designers more available working space because the disk controller no longer requires an expansion slot, but rather, a small interface connector on the motherboard.

Last, many new personal computers have a very high speed I/O bus called the Small Computer System Interface or *SCSI* (pronounced "scuzzy"). The SCSI bus connects disk drives, tape drives, CD-ROMs, and other devices to your system. Because the bus is highly specialized, it provides excellent performance. Before you purchase a SCSI drive you need to ensure that your system has a SCSI bus.

Expansion Slots and Cards

In the back (on the back-left corner in the models pictured) of your chassis you will find several hardware boards and possibly empty slots for more. Typically, your computer will have a disk controller card, additional memory, a video card, and one or more cards containing serial or parallel ports. Serial ports connect printers, a mouse, a modem, or other unique devices. The parallel ports typically only connect printers. Chapter 13 covers the serial and parallel ports.

When you add new hardware to your system, you might need to install a board into one of the expansion slots. To do so, use a screwdriver to remove the metal slot cover, as shown in Figure 1-20.

Place the cover in a safe storage location so you can later cover the slot should you ever remove a board. Next, gently slide the board into the guides as shown in Figure 1-21.

Make sure the board fits completely into the connectors attached to the motherboard. If you don't completely set the board into the slot, the computer will not be aware of the board's presence, or you may experience intermittent errors. Last, secure the board into the slot using the screw you removed with the slot cover.

Always handle computer boards gently during installation. Most boards contain sensitive electronic chips that you can easily damage. Also, make sure you don't loosen another board or cable during your board installation.

NOTE

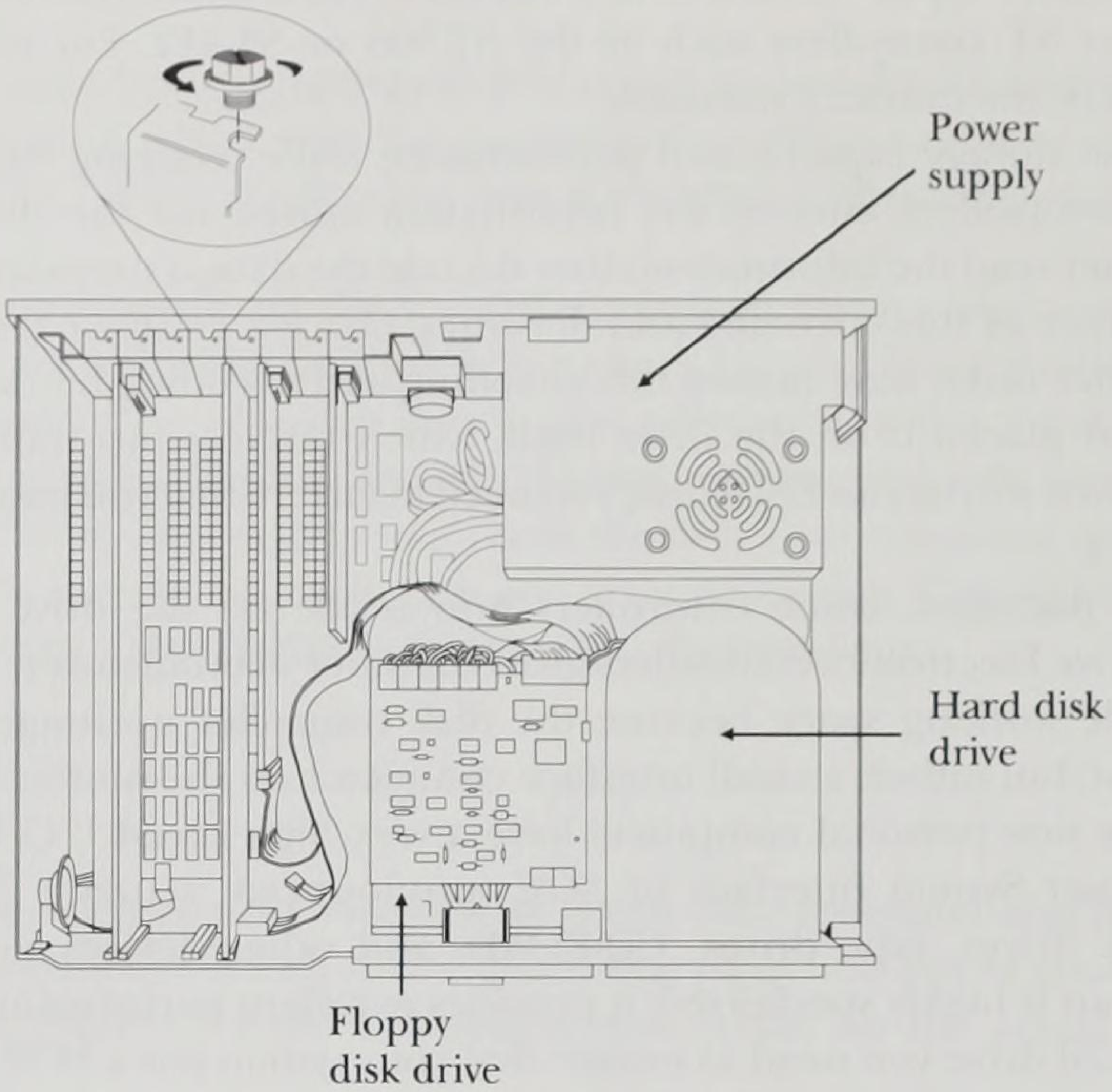

Power supply
Hard disk drive
Floppy disk drive

Figure 1-20. *Removing an expansion slot cover*

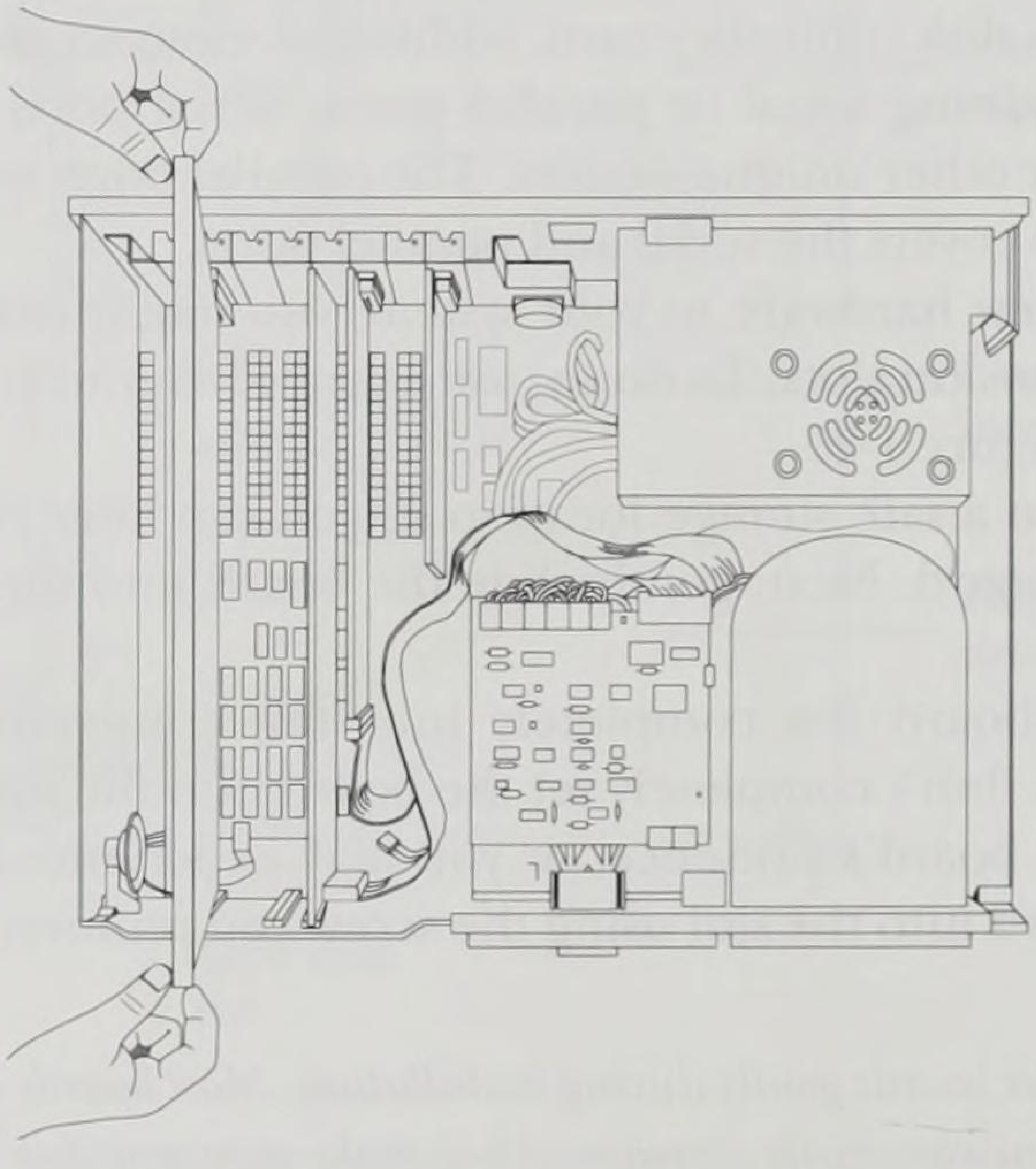


Figure 1-21. *Sliding a board into the expansion slot*

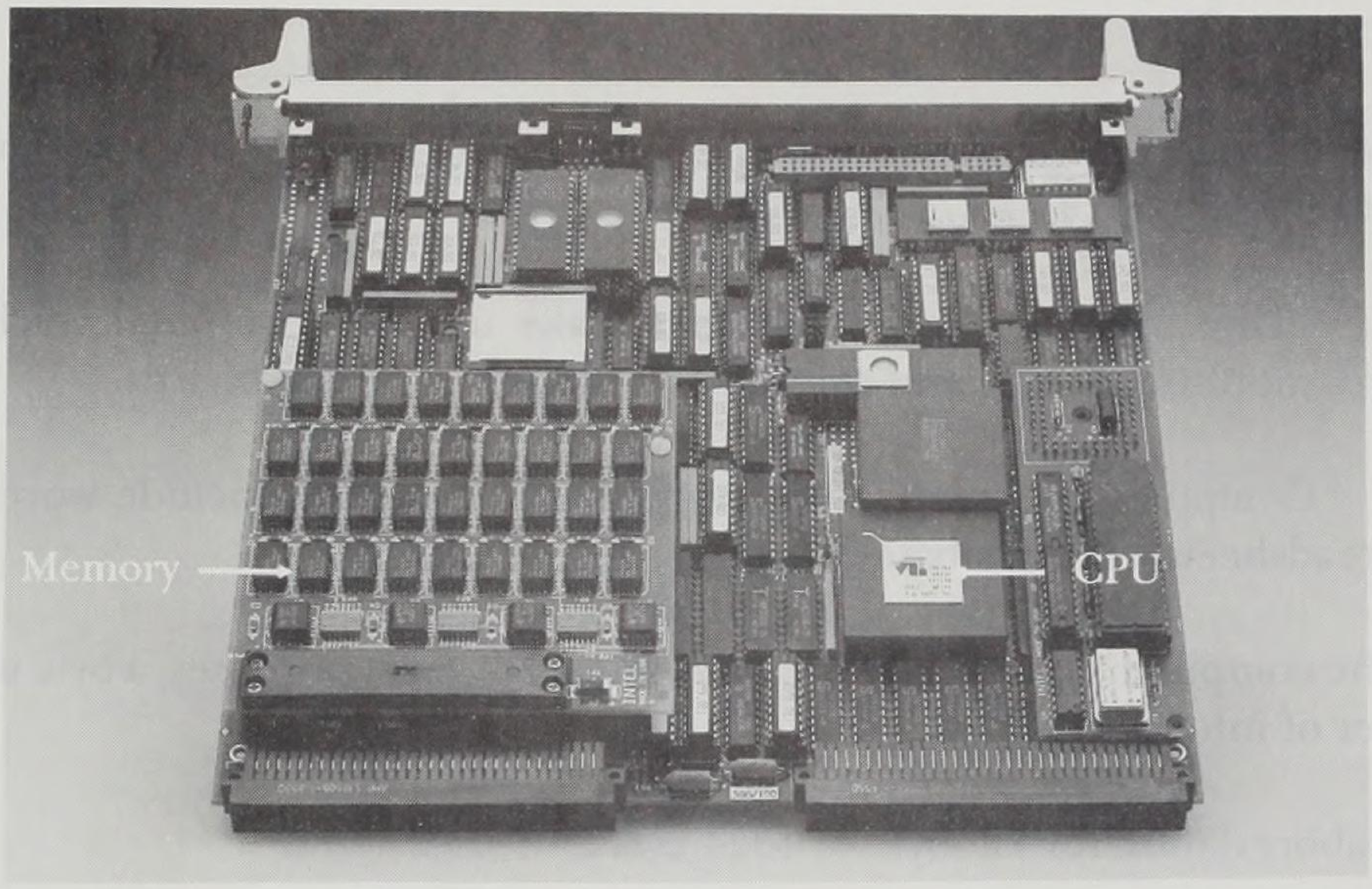


Figure 1-22. *An 80386 motherboard*

Motherboard

Your computer's motherboard is the largest electronic board in the computer. The motherboard holds the computer's central processing unit (CPU), memory, math coprocessor, and other essential chips. In addition, the motherboard has the connectors for all of your expansion boards and controllers. This row of interface connectors is the *backplane.* Figure 1-22 illustrates the motherboard for an 80386.

As you know, computers are categorized by their CPU type, such as 8086, 80286, 80386, and 80486. Table 1-3 describes the essential differences between these processor types.

Processor	Processor Size	Bus Size	Coprocessor
8086	16 bit	16 bit	8087
8088	16 bit	8 bit	8087
80286	16 bit	16 bit	80287
80386SX	32 bit	16 bit	80387SX
80386DX	32 bit	32 bit	80387DX
80486SX	32 bit	32 bit	80487SX
80486DX	32 bit	32 bit	Built-in

Table 1-3. *CPU Types in the IBM PC and PC-Compatibles*

Key Terms

Bus The common internal data channel used by the computer to transfer data to and from the CPU.

Hardware The physical components that make up your computer, such as the keyboard, monitor, chassis, and printer, as well as the internal boards.

Software Computer programs. Common software packages include word processors, spreadsheets, and database programs.

Byte The computer's basic unit for data storage. For most purposes, a byte is a single character of information.

Kb An abbreviation for kilobyte or 1024 bytes.

Mb An abbreviaton for megabyte or 1,048,576 bytes.

Operating system The first program a computer runs, which provides the hardware interface for other application programs. DOS is your computer's operating system. DOS stands for Disk Operating System.

Chapter 2

Getting DOS Started

Chapter 1 introduced you to your computer's hardware, such as the keyboard, monitor, and printer. At that time, you learned that your computer's hardware tells only half the story; software (computer programs) provides the second half. Before you can use your computer to run programs such as your word processor or spreadsheet, your computer must run a special program called the operating system. The operating system serves as your host while you work with your computer. In this chapter you will learn how to start DOS. Once DOS is running, you can type in DOS commands, as discussed in Chapter 3.

Before You Begin

This chapter assumes you have already installed DOS on your hard disk, or your computer is floppy disk based and you will start DOS using a floppy DOS system disk. If you are currently using DOS version 5 or earlier and are planning on swithcing to DOS 6, Appendix D tells you how to perform the upgrade.

Starting DOS from Floppy Disks

If your computer has DOS installed on a hard disk, continue your reading at the section titled "Turning On Your Computer." If you are starting DOS from floppy disks, which is a common practice for older PCs or laptop computers, you will need your DOS system disks. Typically, most floppy disk versions of DOS use two floppy disks, one labeled "Program Disk" and one labeled "Supplemental Programs," as shown in Figure 2-1.

To start DOS from a floppy disk, insert the disk labeled "Program Disk" into drive A and close the disk drive latch as discussed in Chapter 1. You are now ready to continue your reading with the next section.

Turning On Your Computer

If your computer has DOS installed on a hard disk, make sure drive A does not contain a floppy disk and the drive latch is open. As you will learn, your computer always looks to drive A first for a DOS disk and then examines your hard disk. If your floppy disk drive contains a disk other than DOS, your computer might not be able to start DOS.

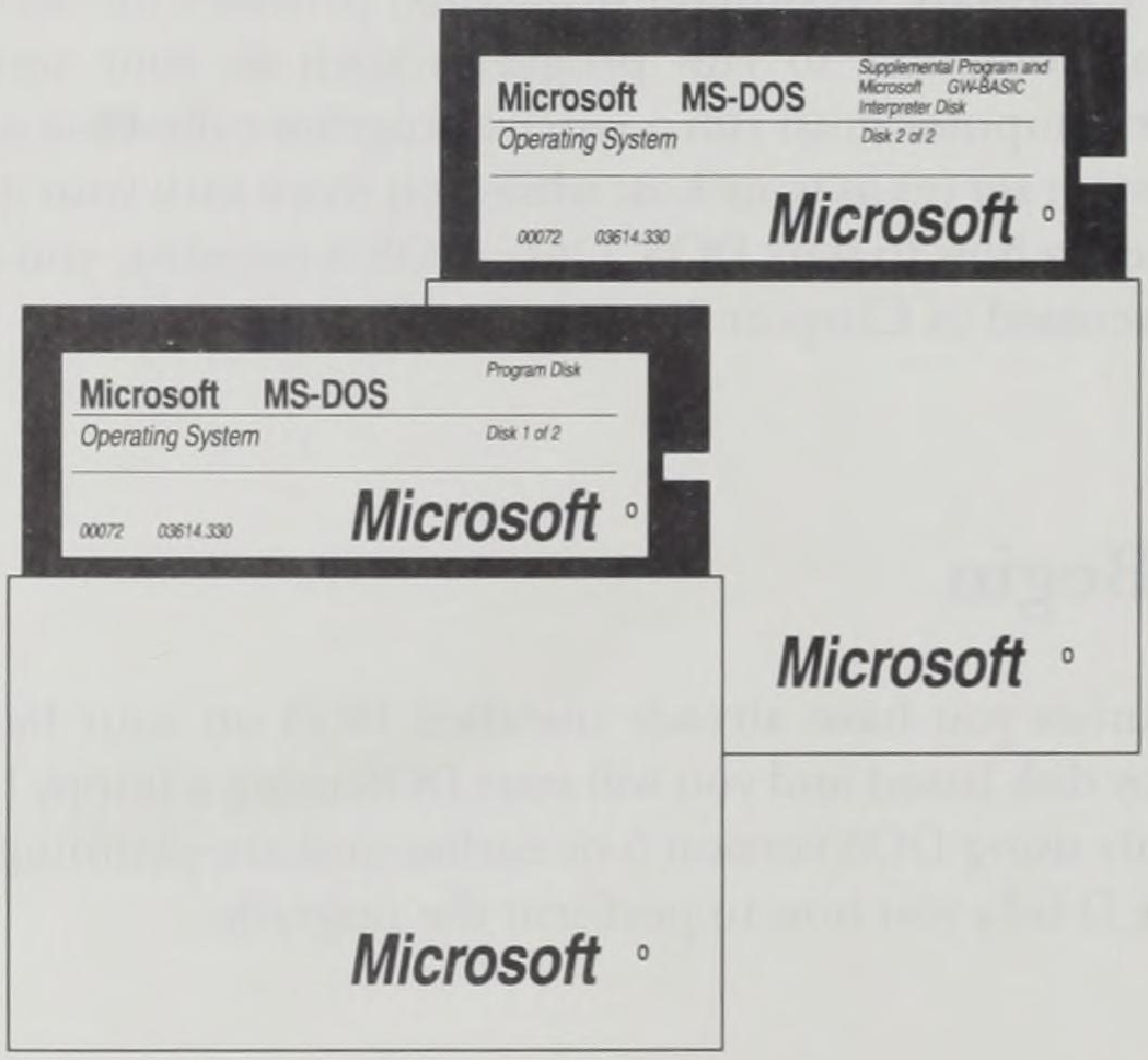

Microsoft MS-DOS
Operating System
Supplemental Program and Microsoft GW-BASIC Interpreter Disk
Disk 2 of 2
00072 03614.330
Microsoft
Microsoft MS-DOS
Operating System
Program Disk
Disk 1 of 2
00072 03614.330
Microsoft
Microsoft
Microsoft

Figure 2-1. *DOS program and supplemental programs floppy disks*

As discussed in Chapter 1, your computer, monitor, and printer each have their own power cables and on/off switches. Make sure all three devices are plugged into working plugs. Next, individually turn on your monitor, computer, and printer. The order in which you turn the devices on does not matter.

When you turn your computer on, you can hear it start to come to life as its internal fan begins to whir. Next, your monitor may display a sequence of increasing numbers as your computer examines its electronic memory. During this startup period, your computer examines many of its critical electronic components. Because your computer only performs these tests when you turn your computer on, the tests are collectively called a *power-on self-test.* Should one of the components fail its power-on test, your computer will likely display an error message you will need to write down and give to your computer repair person.

If your computer successfully passes its power-on self-test, you will see drive A's disk activation light illuminate briefly as your computer searches drive A for a disk containing DOS. If drive A contains a DOS system disk, your computer will begin loading DOS from the floppy disk into your computer's electronic memory.

If drive A does not contain a floppy disk, your computer will load DOS from your hard disk. If the disk in drive A or your hard disk contains DOS, your computer will successfully load DOS. If the disk does not contain DOS, your computer will display an error message (see the troubleshooting tips section later in this chapter).

What DOS Will Display

Depending on your DOS version and system configuration, the actual information DOS will display may differ. In the simplest case, DOS will display characters similar to C> or A> called the *DOS prompt.* In this case, the letter C tells you DOS is currently using drive C, your hard disk. Likewise, the letter A informs you DOS is using the floppy disk in drive A.

Many times, DOS may ask you to set the system date and time before displaying its prompt. In such a case, DOS will prompt you for the date as follows:

```
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy):
```

If DOS displays this prompt, simply press ENTER to continue, leaving the current date unchanged. If the date DOS displays is incorrect, don't worry. Chapter 3 teaches you how to set your computer's date. Next, DOS will prompt you to type in the system time, as shown in the following command.

```
Current time is 6:04:32.32a
Enter new time:
```

Once again, simply press the ENTER key to continue leaving your computer's time unchanged.

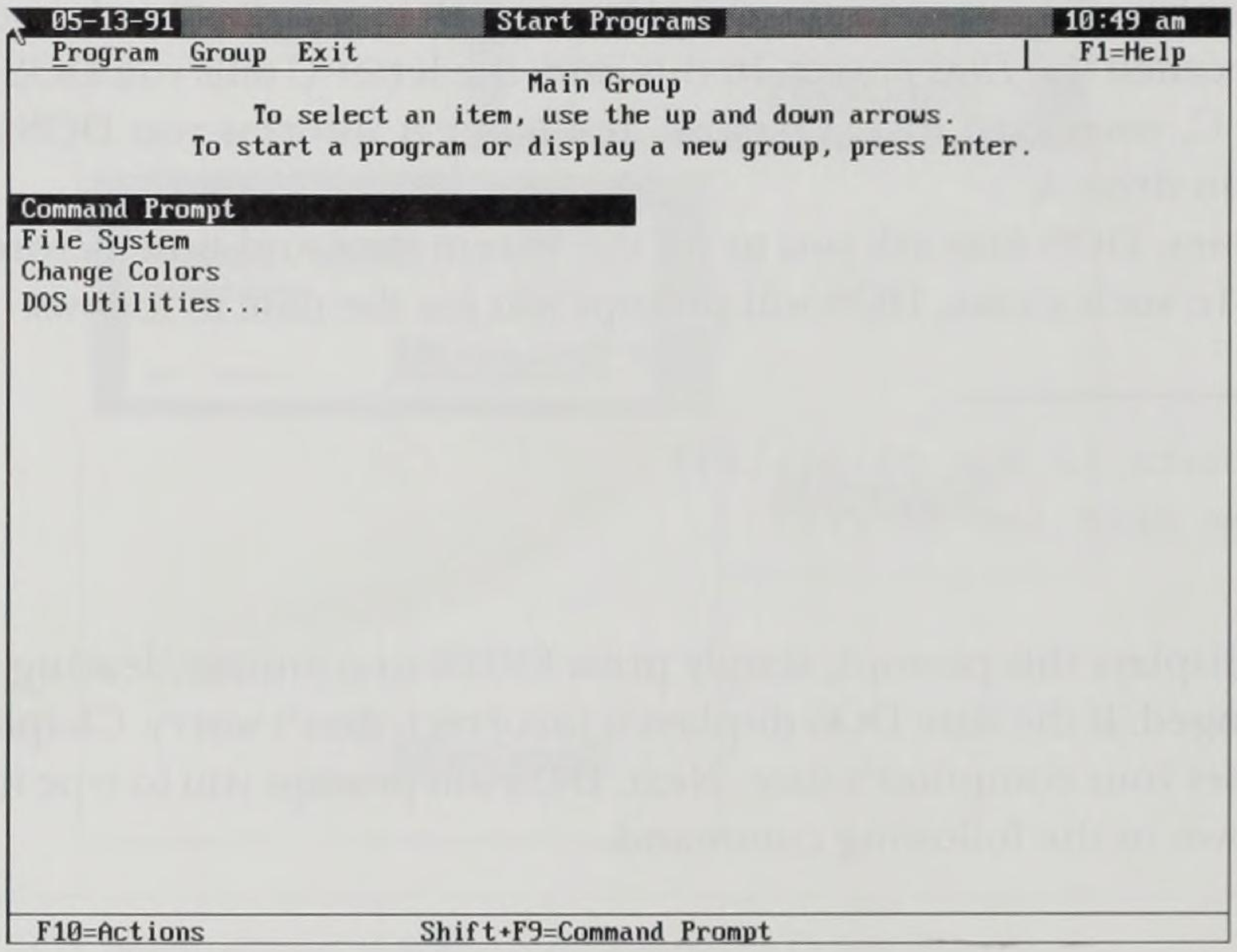
DOS may display its user-friendly shell interface, or possibly Microsoft Windows. Figure 2-2 illustrates the DOS 4 shell.

If DOS displays this shell, hold down your keyboard's ALT key and press the F3 function key to exit the shell to the DOS prompt. Chapter 3 begins its discussion of DOS commands you issue at the DOS prompt.

If you are using DOS 5 or later, DOS may automatically display its shell interface. Figure 2-3 displays the DOS 5 and 6 shell. If DOS displays this shell, press the F3 function key (or press the ALT-F4 key combination) to return to the DOS prompt. Chapters 40 and 41 discuss the DOS 5 and 6 shells in detail.

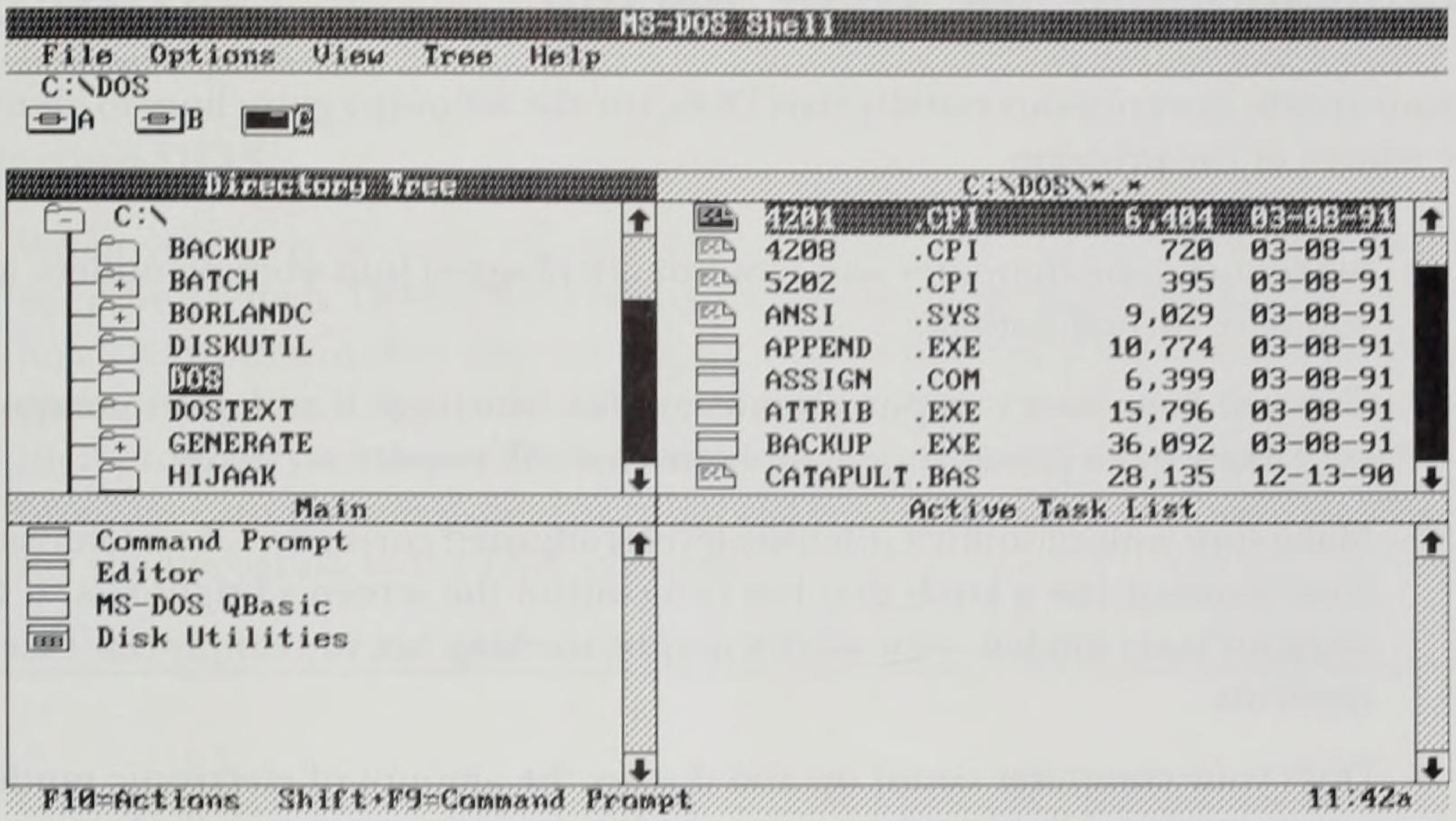
Finally, Figure 2-4 displays the Microsoft Windows interface. If your computer displays Windows, hold down the ALT key and press F to invoke the File menu. Next, press X to select the Exit option, to return to the DOS prompt (or press the ALT-F4 key combination).

Should your computer display a different program, exit the program to the DOS prompt.



```
 05-13-91                  Start Programs                  10:49 am
 Program  Group  Exit                                     | F1=Help
                             Main Group
              To select an item, use the up and down arrows.
          To start a program or display a new group, press Enter.

 Command Prompt
 File System
 Change Colors
 DOS Utilities...

 F10=Actions               Shift+F9=Command Prompt
```

Figure 2-2. *DOS 4 shell program screen*



```
                                 MS-DOS Shell
 File  Options  View  Tree  Help
 C:\DOS
 A   B   C
            Directory Tree                          C:\DOS\*.*
  C:\                                    4201     .CPI      6,404  03-08-91
    BACKUP                               4208     .CPI        720  03-08-91
    BATCH                                5202     .CPI        395  03-08-91
    BORLANDC                             ANSI     .SYS      9,029  03-08-91
    DISKUTIL                             APPEND   .EXE     10,774  03-08-91
    DOS                                  ASSIGN   .COM      6,399  03-08-91
    DOSTEXT                              ATTRIB   .EXE     15,796  03-08-91
    GENERATE                             BACKUP   .EXE     36,092  03-08-91
    HIJAAK                               CATAPULT.BAS      28,135  12-13-90
                 Main                             Active Task List
  Command Prompt
  Editor
  MS-DOS QBasic
  Disk Utilities

 F10=Actions  Shift+F9=Command Prompt                                11:42a
```

Figure 2-3. *DOS 5 and 6 shell program screen*

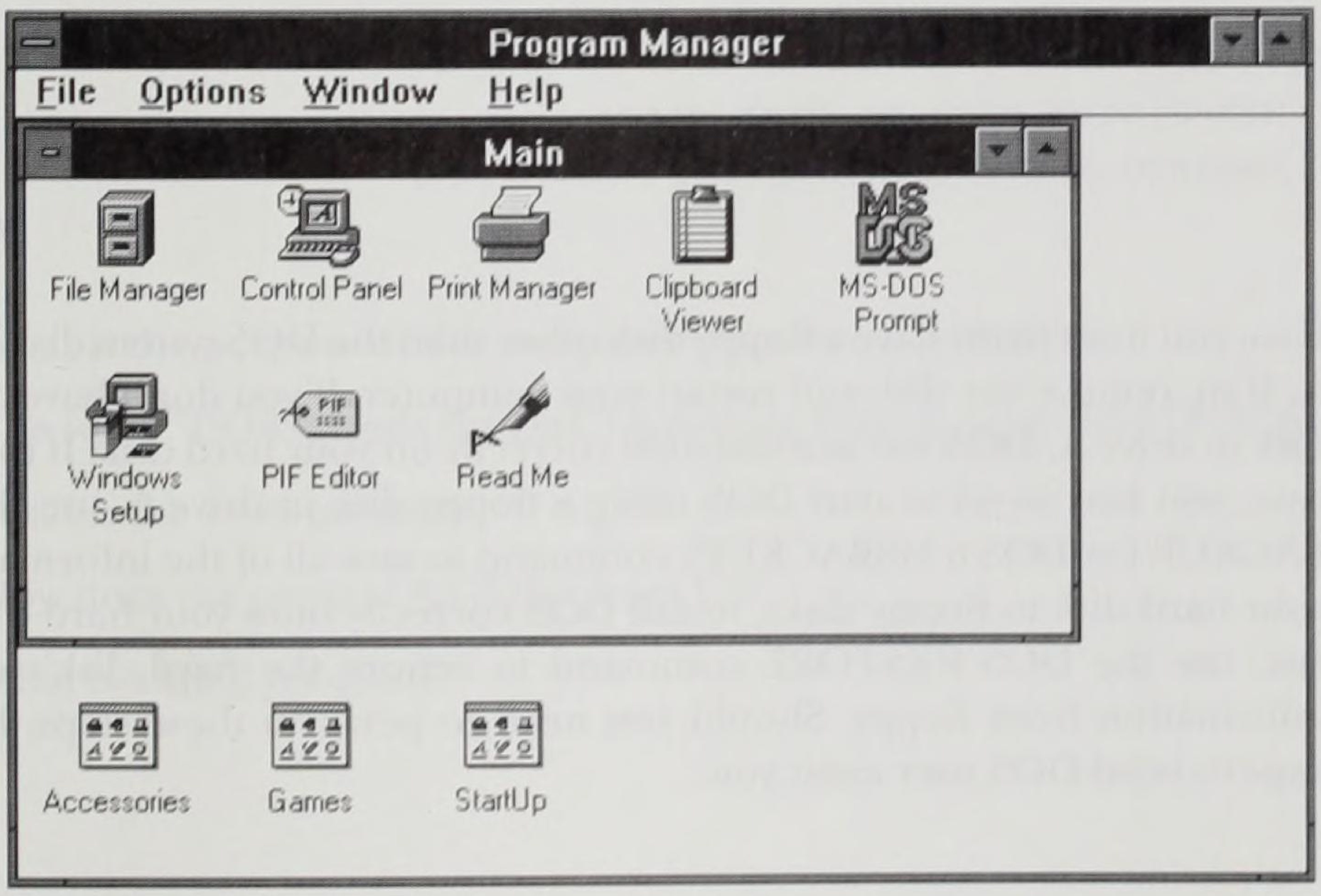

Program Manager
File Options Window Help
Main
File Manager
Control Panel
Print Manager
Clipboard Viewer
MS-DOS Prompt
Windows Setup
PIF Editor
Read Me
Accessories
Games
StartUp

Figure 2-4. *Microsoft Windows interface*

Troubleshooting the DOS Startup

If your system does not successfully start DOS, use the following guidelines to identify the source of the problem.

1. Make sure your computer and monitor are plugged into working outlets, and you have turned both on.
2. Can you hear your computer's internal fan whirring? If not, your computer very likely has a power supply problem that will require servicing.
3. Make sure your monitor's intensity level is adjusted correctly. Like a television, your monitor has a knob that lets you control the screen's brightness. If the intensity is set too low, your screen may be working but you simply can't see its contents.
4. Does your computer count up and display the amount of electronic random access memory? During the power-on self-test, most computers display a count of its working memory. If yours does not, the computer may be failing its self-test very early in the test sequence.
5. Does your computer illuminate drive A's disk activation light in search of DOS? As discussed, after your computer finishes its power-on self-test, the computer will first look for DOS on a floppy disk in drive A. If your computer does not, it is failing its self-test.
6. Does your screen display the following error message?

```
Non-System disk or disk error
Replace and strike any key when ready
```

   If so, you most likely have a floppy disk other than the DOS system disk in drive A. If so, remove the disk and restart your computer. If you don't have a floppy disk in drive A, DOS was not installed correctly on your hard disk. If this is the case, you may need to start DOS using a floppy disk in drive A, use the DOS BACKUP (or DOS 6 MSBACKUP) command to save all of the information on your hard disk to floppy disks, install DOS correctly onto your hard disk, and last, use the DOS RESTORE command to restore the hard disk's original information from floppy. Should you need to perform these steps, have an experienced DOS user assist you.

SUMMARY

Starting DOS

When you turn on your PC, the first program that runs is DOS, the disk operating system. DOS in turn lets you run other programs. Depending upon how DOS was installed on your system, DOS may display its command-line prompt (such as C:\>), prompt you for the current date and time, or run a shell program such as the DOS 5 and 6 shell or Microsoft Windows. If a shell program runs on your system, you can normally select a menu option to exit the shell program to the DOS prompt.

Hands On

Beginning with Chapter 1, the chapters contain a "Hands On" section, which lets you experiment with several different DOS commands. For now, start DOS following the steps that have been presented. If DOS displays a shell or menu program similar to those discussed, exit the program to the DOS prompt.

Review

1. What is the purpose of the computer's power-on self-test?
2. The word "DOS" stands for Disk Operating System; what does DOS do?
3. What is the DOS prompt?
4. How does the prompt A> differ from C>?
5. What is a shell program?

Answers

1. Each time you turn on your computer's power, the computer performs a series of diagnostic tests to ensure that all of its electronic components are working properly. Although computers are very dependable, they do eventually wear out. The computer's self-test examines the key internal electronic components. Hopefully, if a component fails, your computer can display a message indicating the cause and location of its failure.
2. The DOS operating system serves as your host while you work at the computer. Operating systems provide your interface to the computer's hardware and let you run other computer programs such as your word processor or spreadsheet.
3. When you run programs at your computer, you type in the program's name and press ENTER. The DOS prompt is a series of one or more letters that DOS displays to inform you it is ready for you to type in commands. Typically, the DOS prompt contains the letter A for floppy disk systems (perhaps A>) or the letter C for hard disk systems (C>). Regardless of the letters used, the DOS prompt is simply your notification that DOS is ready for you to issue commands.
4. The DOS prompt typically contains the letter that corresponds to the current disk drive. The A> prompt indicates that DOS is currently using floppy disk drive A. Likewise, the C> prompt indicates DOS is using the hard disk drive C. Chapter 6 examines using your computer's disk in detail.
5. A shell program is a menu-driven user-friendly interface that makes the computer easier to use. From within a shell program, users can run other programs, copy files, and perform many of the same tasks provided by DOS. Shell programs have grown in popularity due to their ease of use. DOS versions 4 and later provide shell interfaces.

Advanced Concepts

Although DOS is the most widely used operating system in the world, and millions of users start DOS every day, few users understand the DOS startup sequence. In this section we will examine this process in detail.

How Your Computer Finds DOS

As you know, each time you start your computer, your computer first looks for DOS on a floppy disk in drive A, regardless of whether or not your computer has a hard disk with DOS installed. Here's why.

First, hard disks are not shipped from the disk manufacturers with DOS installed, as they may supply drives to companies using other operating systems. Instead, the hard disk must be installed in a computer and formatted as a bootable disk. To format your hard disk, you need a way of initially starting DOS other than your hard disk. As such, your computer always looks first to drive A.

Second, although hard disks are very reliable, they eventually wear out or become damaged. Should your hard disk fail, you need a way to start your computer.

Last, users periodically inadvertently issue commands that delete or rename key files that DOS needs to start. By letting the user boot DOS from a floppy, the user can get DOS running and normally correct the error.

If you are using a hard disk, your computer must locate DOS on one of your hard disk partitions. As you will recall, during the DOS installation process, the FDISK command lets you divide your disk into one to four *partitions.* DOS views each partition as a unique disk drive. Assuming your hard disk has two partitions, DOS would access the first partition as drive C, and the second partition as drive D. Using FDISK, you must select one partition as the primary partition from which your computer will boot DOS. The first thing your computer must do when it tries to boot DOS from your hard disk is determine the primary DOS partition. To do so, your computer reads the disk's master boot record.

As part of its processing, the FDISK command writes information that describes the disk's partition to the very first sector of your disk. This information, called the *master boot record,* tells your computer where each partition begins and the partition's size. It also identifies the primary DOS partition, from which your computer will boot DOS. Figure 2-5 illustrates a disk with two partitions, as well as the master boot record.

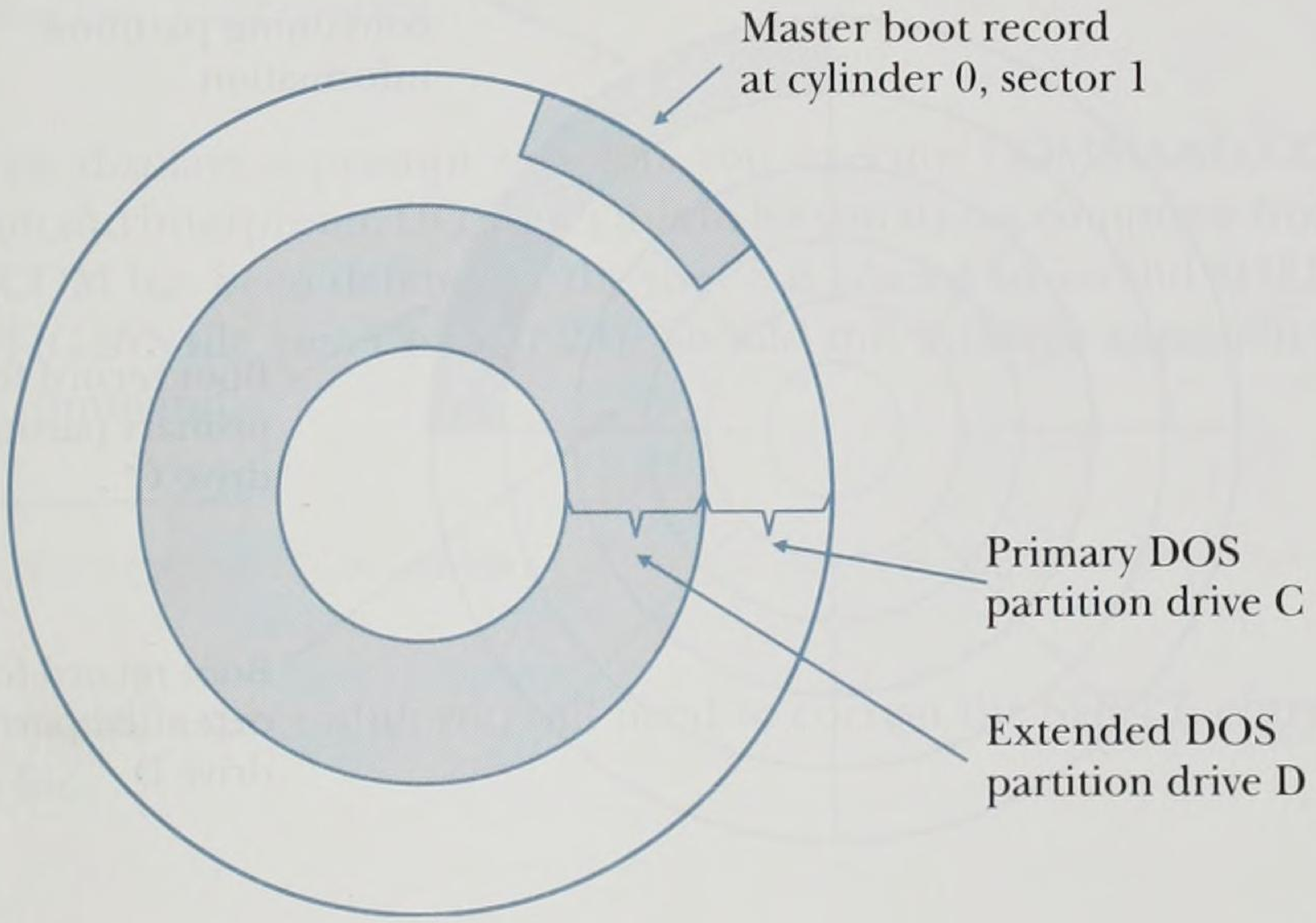


Figure 2-5. *A two-partition disk*

Using the information it finds in the master boot record, your computer locates the primary DOS partition and reads its first sector, the DOS boot record. As you just read, DOS views each partition as a unique disk. The DOS boot record describes the boot disk characteristics, such as the sector size, number of sectors, and so on. In addition, the boot record contains a small program the computer loads into memory to start the DOS boot process. Figure 2-6 illustrates the DOS boot record in relation to the disk's primary DOS partition.

Every bootable DOS disk has two hidden files. For MS-DOS, these two files are named IO.SYS and MSDOS.SYS. Likewise, PC-DOS names these files IBMBIO.COM and IBMDOS.COM. The function of the MS-DOS and PC-DOS files are the same. The following discussion uses the MS-DOS filenames. The file IO.SYS contains software that builds upon your computer's basic I/O capabilities as provided in ROM. The file MSDOS.SYS contains the software that implements the DOS system services. DOS uses these services for a wide range of tasks such as reading and writing to disk, your printer, or even setting your computer's date and time. If these two files are not present, DOS will display the following error message:

```
Non-System disk or disk error
```

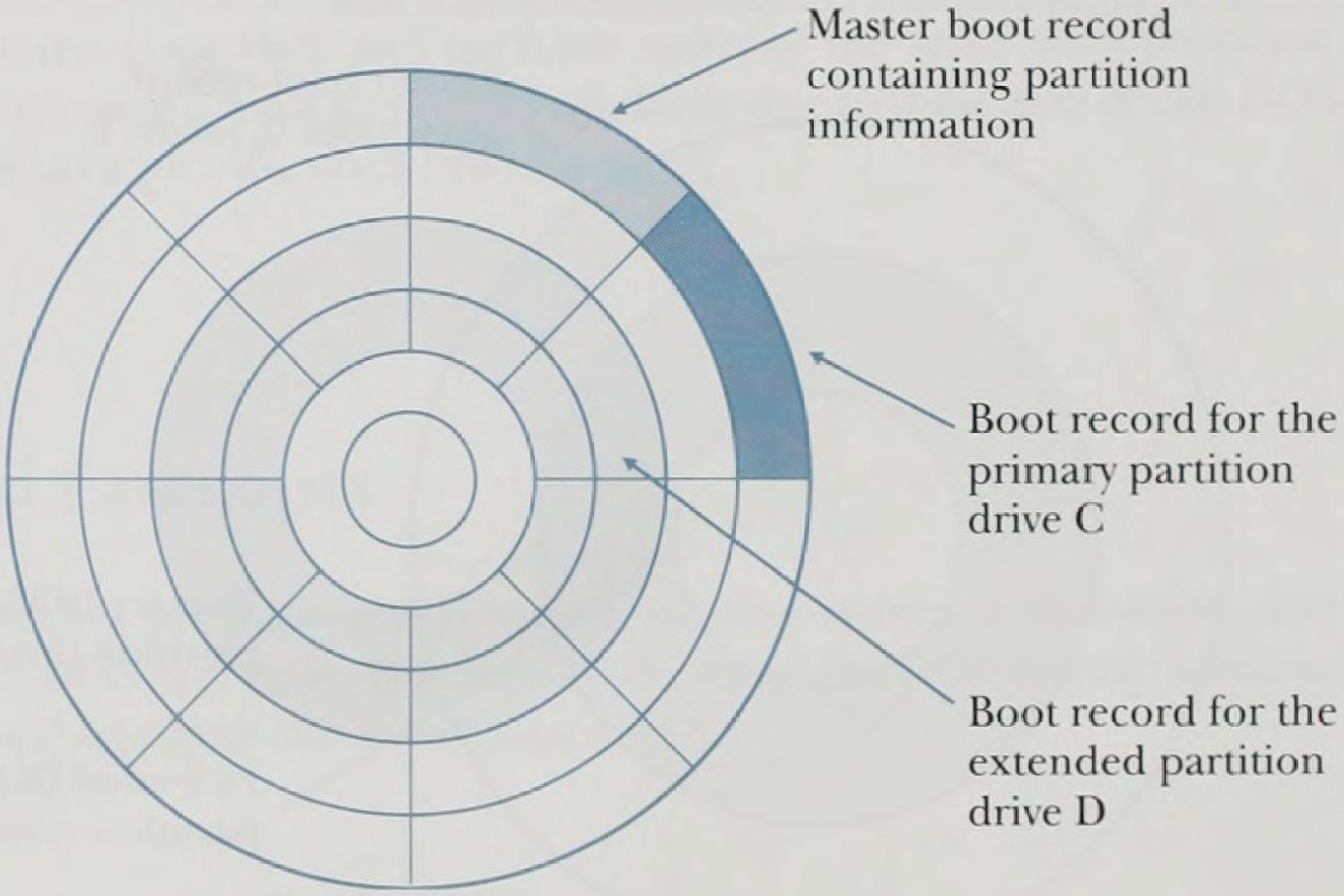

Master boot record containing partition information
Boot record for the primary partition drive C
Boot record for the extended partition drive D

Figure 2-6. *DOS boot record within the primary DOS partition*

Assuming the hidden files exist, DOS first loads the file IO.SYS into memory. In addition to containing the additional I/O services, IO.SYS contains a small system initialization program named SYSINIT. When SYSINIT is in memory, DOS invokes it, and SYSINIT begins reading the file MSDOS.SYS from disk into memory. Next, SYSINIT searches the disk's root directory for the file CONFIG.SYS. SYSINIT uses the CONFIG.SYS entries to configure itself in memory. Last, SYSINIT locates and invokes the file COMMAND.COM, which contains all of the internal DOS commands as well as the DOS command processor responsible for displaying the DOS prompt and processing your commands. If SYSINIT cannot locate COMMAND.COM, the system startup will stop and DOS will display the following error message:

```
Bad or missing Command Interpreter
```

If this occurs, you may need to start DOS from a floppy disk and copy COMMAND.COM into your hard disk's root directory. Assuming SYSINIT finds and loads it, COMMAND.COM, in turn, searches your root directory for the file AUTOEXEC.BAT. If AUTOEXEC.BAT exists, COMMAND.COM will invoke it. Otherwise, COMMAND.COM will invoke the familiar DATE and TIME commands completing the startup or boot process.

In DOS 6, if SYSINIT cannot locate COMMAND.COM, it will display the following message:

```
Bad or missing Command Interpreter
Enter correct name of Command Interpreter
   (for example, C:\COMMAND.COM)
```

DOS 6 then displays a prompt that lets you execute COMMAND.COM or a different command interpreter. DOS does this to let you start a computer from which COMMAND.COM has been deleted or the user has placed an invalid SHELL entry in their CONFIG.SYS file (see Chapter 24). Should this message appear, try issuing the following command:

```
C:\> C:\DOS\COMMAND.COM <ENTER>
```

If the command is successful, you will need to correct the SHELL entry in your CONFIG.SYS file.

Key Terms

Booting DOS Starting DOS. The process of loading DOS from disk into your computer's electronic memory.

Non-system disk A disk that does not contain the files needed to start DOS.

System disk A disk containing the DOS hidden files used to load DOS and the DOS command processor COMMAND.COM.

Disk partition A portion of a hard disk for use as a logical drive.

Primary DOS partition The hard disk partition containing the system information used to start DOS.

DOS prompt The characters DOS displays to indicate it is waiting for you to enter a command. Common DOS prompts include C> and A>.

Chapter 3

Issuing DOS Commands

In Chapter 2 you learned that after DOS successfully loads itself from disk into your computer's memory, it displays its *system prompt,* which indicates that DOS is ready for you to type in a command. In this chapter you will execute your first DOS commands. As you will learn, to improve its performance, DOS divides its commands into two types: internal commands, which are stored in your computer's fast electronic memory, and external commands, which are stored on disk. This chapter helps you understand why DOS does this, and how it affects the steps you must follow to execute each command type successfully.

Executing Your First DOS Commands

As you read in Chapter 2, your system prompt may differ from the C> prompt used throughout this chapter. Because most DOS users today use the hard disk drive C, this chapter uses the drive letter within the prompt. If your system prompt differs, don't worry. The prompt will not change the way DOS executes the example commands. Don't worry if your prompt doesn't show the backslash either (for example, C:\). You can learn more about the prompt in Chapter 10.

To execute a DOS command, you type the command's name at the DOS prompt and press ENTER. DOS will execute the command, redisplaying its prompt after the command completes so you can type in your next command. First, if you simply press the ENTER key without typing a command name, DOS will redisplay its prompt

```
C> <ENTER>
C> <ENTER>
C>
```

CLS Clears the Screen

As you work with DOS, your screen can periodically become cluttered with the output of your previous commands. The CLS command lets you erase or clear your screen contents. DOS abbreviates many of its commands to reduce your typing. CLS, for example, is an abbreviation for clear screen. To erase your screen's contents, type **CLS** and press ENTER.

```
C> CLS <ENTER>
```

DOS will erase your screen's contents and redisplay its prompt in the upper-left corner of your screen (your screen's home position).

VER Displays the DOS Version Number

When Microsoft first released DOS in 1981, DOS initially had the version number 1.0. Over the years, as Microsoft has increased the capabilities DOS provides, they have released newer, more powerful versions of DOS. Each time the DOS developers release a new version, they increment the DOS version number. VER displays your DOS version number on the screen. To execute VER, type **VER** and press ENTER.

```
C> VER <ENTER>
```

VER will respond by displaying the DOS version number. In DOS 3.3, VER will display the following:

```
MS-DOS Version 3.30
```

Likewise, in DOS 6, VER displays the version number:

```
MS-DOS Version 6.00
```

You must know your DOS version number. Some DOS commands are available in one version of DOS and not others. Likewise, many applications, such as your word processor or spreadsheet, may require a specific version of DOS. For the purposes of this book, however, unless you are explicitly told otherwise, you can assume the concepts discussed apply to every version of DOS.

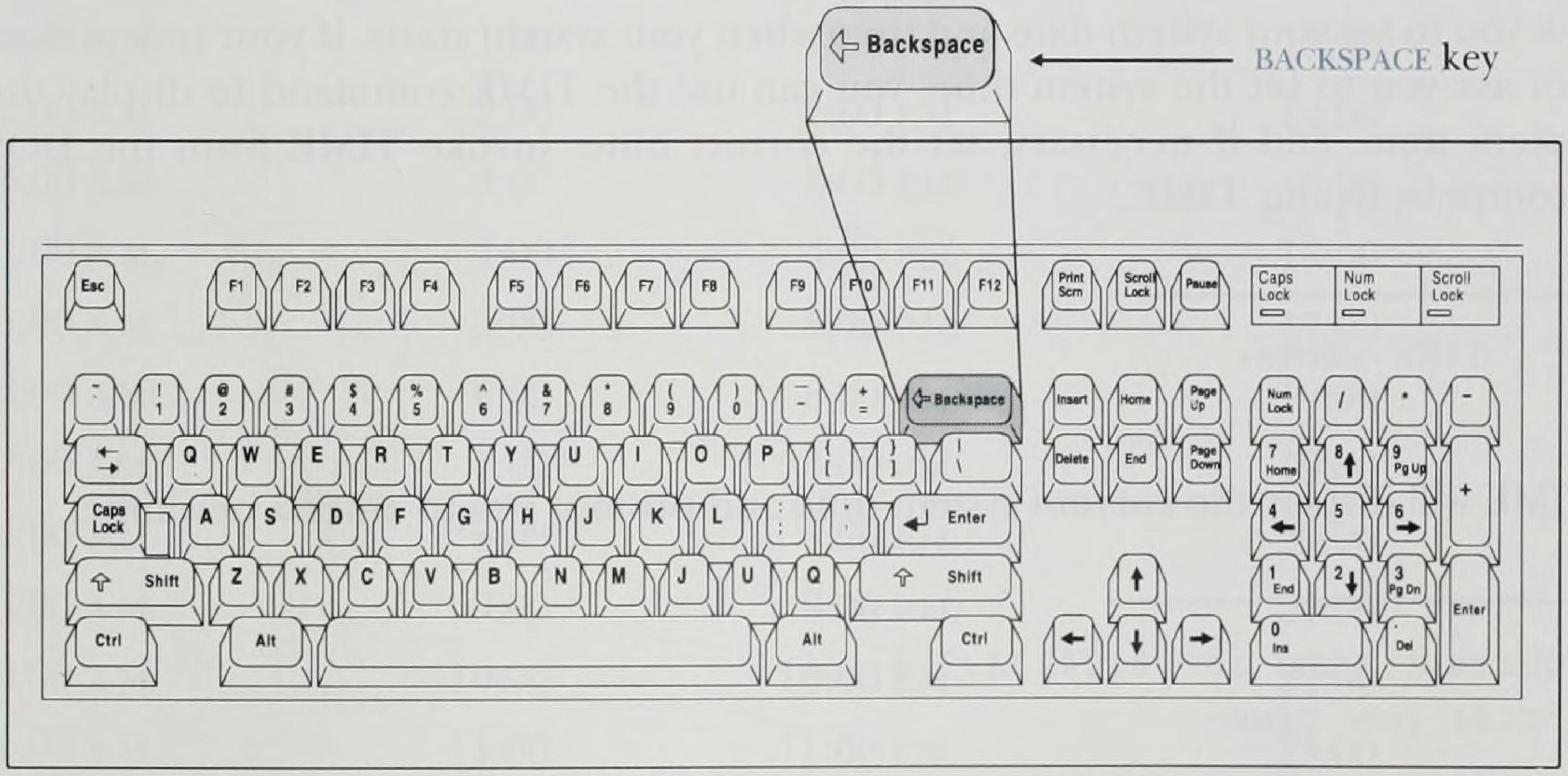
You have learned that, to execute a DOS command, you type the command's name at the DOS prompt and press ENTER. If, before you press ENTER, you notice that you have mistyped one or more letters in the command's name, you can correct the command using your keyboard's BACKSPACE key, shown on one common keyboard in Figure 3-1.

Suppose, for example, you mistyped the CLS command as "CKS" but have not yet pressed ENTER:

```
C> CKS_
```

When you press BACKSPACE, DOS will first erase the S:

```
C> CK_
```



BACKSPACE key

Figure 3-1. *Location of the keyboard's* BACKSPACE *key*

By pressing BACKSPACE a second time, the K is erased.

```
C> C_
```

Next, retype the letters L and S and press ENTER:

```
C> CLS <ENTER>
```

Upper- or Lowercase Letters

For consistency, all the command examples presented throughout this book use uppercase letters. DOS, however, does not care if you type your command names in upper- or lowercase letters, or a mixture of both. To DOS, the following three commands are all the same.

```
C> CLS <ENTER>

C> cls <ENTER>

C> cLs <ENTER>
```

TIME Displays or Sets the System Time

You learned in Chapter 2 that, depending on your system configuration, DOS may ask you to set your system date and time when your system starts. If your system does not ask you to set the system time, you can use the TIME command to display the system time, and if necessary, set the correct time. Invoke TIME from the DOS prompt by typing TIME.

```
C> TIME <ENTER>
```

TIME will display the current system time and prompt you to enter a new time:

```
Current time is  6:04:32.32a
Enter new time:
```

If you simply press ENTER, TIME will leave the current system time unchanged. Many users will invoke TIME throughout the day, simply to determine the current time. By pressing ENTER at TIME's prompt for a new time, the current system time is left unchanged.

To set your current system time, you must invoke TIME from the DOS prompt, as before. When TIME prompts you to enter a new system time, you must type in the time desired, specifying hours and minutes in the form *HH:MM*. For example, you would type **10:30** for 10:30 A.M.

DOS versions prior to DOS 4 use a 24-hour military clock, meaning you must add 12 to each hour after noon. 1:00 P.M. becomes 13:00. Likewise, 4:30 P.M. becomes 16:30. Table 3-1 provides standard times and their 24-hour equivalents.

The following TIME command, for example, will set the current system time to 4:30 P.M. (16:30):

```
C:\> TIME <ENTER>
Current time is  6:04:32.32a
Enter new time: 16:30 <ENTER>
```

Likewise, the following command (on the next page) sets the time to 8:30 A.M.

Time	24-Hour Time	Time	24-Hour Time
12 Midnight	0:00	12 Noon	12:00
1:00 A.M.	1:00	1:00 P.M.	13:00
2:00 A.M.	2:00	2:00 P.M.	14:00
3:00 A.M.	3:00	3:00 P.M.	15:00
4:00 A.M.	4:00	4:00 P.M.	16:00
5:00 A.M.	5:00	5:00 P.M.	17:00
6:00 A.M.	6:00	6:00 P.M.	18:00
7:00 A.M.	7:00	7:00 P.M.	19:00
8:00 A.M.	8:00	8:00 P.M.	20:00
9:00 A.M.	9:00	9:00 P.M.	21:00
10:00 A.M.	10:00	10:00 P.M.	22:00
11:00 A.M.	11:00	11:00 P.M.	23:00

Table 3-1. *Standard Times and their 24-Hour Equivalents*

```
C:\> TIME <ENTER>
Current time is 16:30:33.12
Enter new time: 8:30 <ENTER>
```

If you are using DOS 4 or later, DOS gives you an alternative to 24-hour military time and lets you instead append the letter A or P to your time to represent A.M. or P.M. The following command uses this format to set the system time to 2:30 P.M.

```
C:\> TIME <ENTER>
Current time is  8:30:32.32a
Enter new time: 2:30P <ENTER>
```

Correcting an Invalid Time

If you mistype the system time and enter an invalid time, TIME will display an "Invalid time" error message and prompt you to enter a correct time, as follows:

```
C:\> TIME <ENTER>
Current time is  8:30:32.32a
Enter new time: 230P <ENTER>

Invalid time
Enter new time:
```

Should the "Invalid time" message appear, make sure you are specifying the desired time in the form *HH:MM.*

Specifying a More Precise Time

All of the TIME command examples presented so far have only set the hours and minutes. TIME, however, lets you specify hours, minutes, seconds, and even hundredths of a second. The complete format for the TIME command becomes the following: *HH:MM:SS.hh,* where *SS* represents seconds from 0 to 59 and *hh* represents hundredths of a second from 0 to 99. The following TIME command, for example, sets the system time to 11:59:59.99 A.M.

```
C:\> TIME <ENTER>
Current time is  8:30:32.32a
Enter new time: 11:59:59.99 <ENTER>
```

If you don't specify values for the seconds and hundredths of seconds, TIME uses the default value, 0.

DATE Displays or Sets the System Date

As was the case with the system time during startup, DOS may, depending on your system configuration, display the current system date and prompt you to enter a new date. The DATE command lets you display and set the current system date. To display the current system date, type DATE from the DOS prompt:

```
C> DATE <ENTER>
```

DATE will display the current date and prompt you to enter a new date:

```
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy):
```

To leave the date unchanged, just press ENTER. To change the system date, type in the numeric representation of the desired date in the form *mm-dd-yy*, where *mm* is the current month from 1 to 12, *dd* is the day from 1 to 31, and *yy* is the last two digits of the year, such as 93 (optionally, you can include all four digits of the year). For example the following DATE command sets the system date to December 25, 1993.

```
C:\> DATE <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): 12-25-93 <ENTER>
```

You must specify the month, day, and year. If you enter an invalid date, DATE will display an "Invalid date" message and prompt you to enter a new date:

```
C:\> DATE <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): 25-12-93 <ENTER>

Invalid date
Enter new date (mm-dd-yy):
```

If this error message occurs, make sure you are specifying the date in the correct order, and that you are separating each field with a slash (/) or hyphen (-). The date format used throughout this book is *mm-dd-yy,* the format used in the United States. If you have selected a different international system configuration using the CONFIG.SYS COUNTRY= entry, your date format may differ.

Note to Pre-DOS 3.3 Users

If you are using DOS version 3.2 or earlier, the DATE and TIME commands do not permanently set the system clock on the IBM PC AT. Instead, each time your system starts, you must use DATE and TIME to set the system clock. However, if you are using DOS 3.3 or later, the date and time you set remain in effect until you change them at a future time. To set the system clock on a PC AT permanently in DOS 3.2 or earlier, you can use the SETUP disk in your computer's guide to operations, which should have accompanied your computer.

Command-Line Parameters

You have learned that, to execute a DOS command, you type the command's name and press ENTER. The line of information you type before pressing ENTER is the *command line.* Many of the DOS commands you will use throughout this book let you specify additional information in the command line, such as a filename, or even the desired date or time.

You have learned that if you type **TIME** and press ENTER, TIME will display the current time and prompt you to enter a new time. If you want to set the time without viewing the current time, you can include the desired time in the command line, as shown here:

```
C> TIME  11:30 <ENTER>
C>
```

In this case, because your command line contains a valid time, TIME does not display the current time and prompt—it sets the system clock to the time specified.

You can specify the desired date in the command line. For example, the following command sets the system date to October 31, 1993:

```
C> DATE  10-31-93 <ENTER>
C>
```

As before with TIME, DATE suppresses its display of the current date, as well as its prompt for a new date, because the command line contains a valid date.

External Versus Internal Commands

Before any program (or DOS command) can run, the program must first be stored in your computer's electronic memory. Each time you start your computer, for example, your computer must load DOS from disk into its memory chips before running DOS, as shown in Figure 3-2.

Because DOS has so many commands, and your computer has a limited amount of electronic memory (or RAM), DOS only keeps a few of the smallest and most commonly used commands in memory at all times. DOS can execute these commands, called *internal commands,* without first having to load them from disk into memory. As a result, these commands run almost instantly. CLS, VER, DATE, and TIME are all examples of internal commands.

Most DOS commands, however, are *external commands,* which reside on disk. When you execute an external command, DOS must locate the command on disk and then load the command into memory before the command can run, as illustrated in Figure 3-3. All applications, such as word processors or spreadsheets, are external commands.

You may be wondering why you should care about the external and internal commands. The reason has to do with the steps you must follow to execute a command. Because internal commands are always in your computer's memory, you can execute an internal command at any time by typing the command's name. With external commands, however, DOS must locate the command on disk before the command can execute. Unfortunately, DOS, by default, only looks for the command in certain locations. In Chapter 4, you will learn how DOS stores information on your

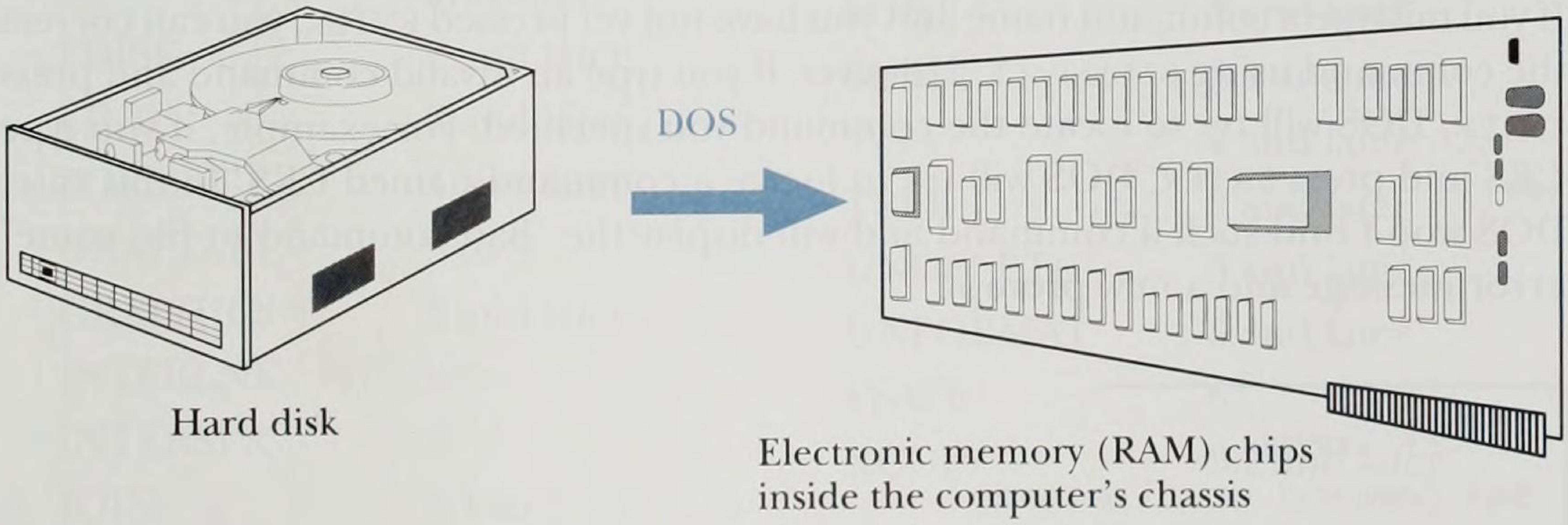


Figure 3-2. *Each time your computer starts, DOS is loaded from disk into the computer's fast electronic memory*

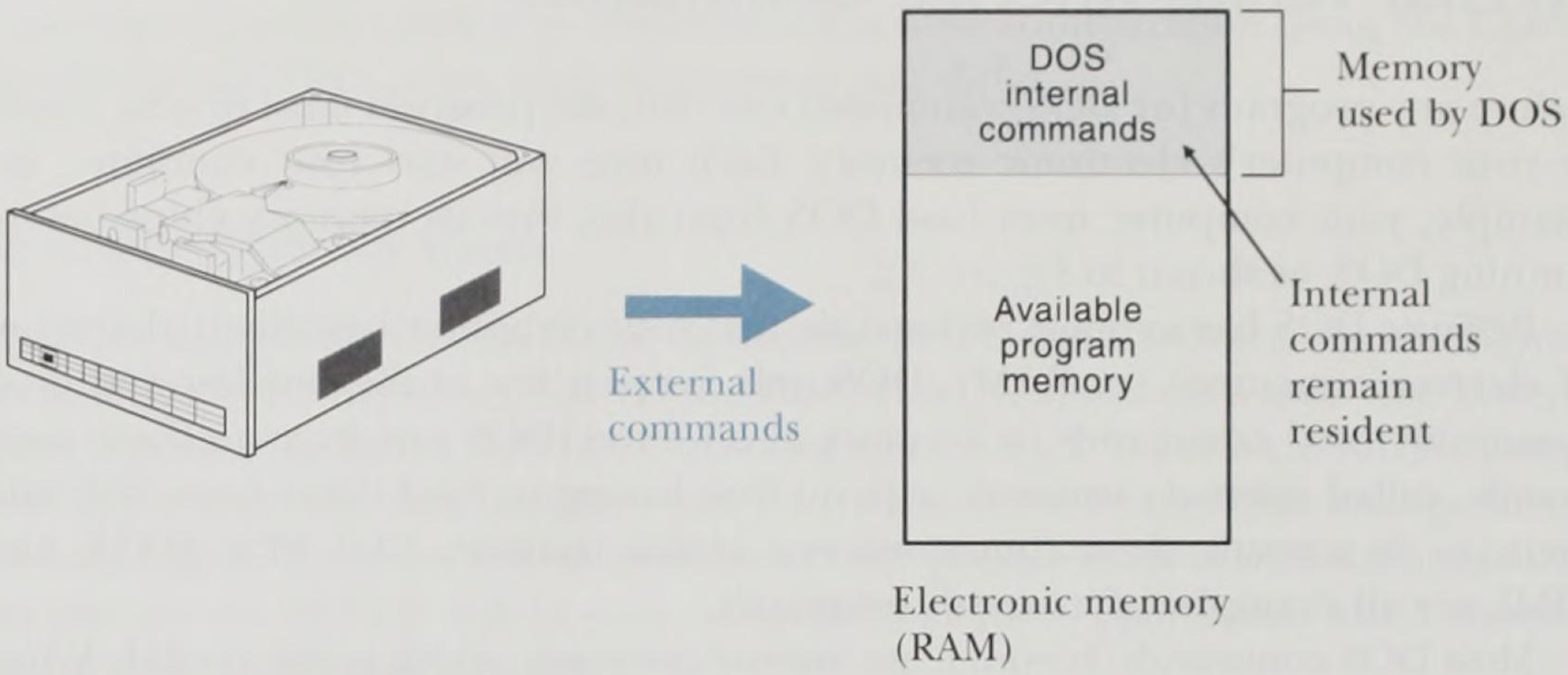

DOS internal commands
Memory used by DOS
Internal commands remain resident
Available program memory
External commands
Electronic memory (RAM)

Figure 3-3. *To execute an external command, DOS must first load the command from disk into memory*

disk using files. At that time, you will begin to understand why it is sometimes difficult for DOS to locate your external commands.

For now, however, simply be aware that internal DOS commands are always in your computer's memory once DOS is active, while external commands are located on disk. Before DOS can execute an external command, DOS must find the command on disk and load the command into memory.

Table 3-2 lists the external DOS commands. Likewise, Table 3-3 lists the internal DOS commands.

The "Bad command or file name" Message

If you mistype a command name and you have not yet pressed ENTER, you can correct the command using BACKSPACE. However, if you type an invalid command and press ENTER, DOS will try to locate the command you specified. For example, if you type **CKS** and press ENTER, DOS will try to locate a command named CKS. In this case, DOS won't find such a command and will display the "Bad command or file name" error message and a new prompt:

```
C> CKS <ENTER>
Bad command or file name

C>
```

Command	DOS Version	Command	DOS Version
APPEND	3.2 and later	KEYB	3.3 and later
ASSIGN	2 to 5	KEYB*xx*	3 to 3.2
ATTRIB	3 and later	LABEL	3 and later
BACKUP	2 to 5	LOADFIX	5 and later
CHDKSK	1 and later	LOADHIGH	5 and later
CHOICE	6	MEM	4 and later
COMMAND	1 and later	MEMMAKER	6
COMP	1 to 5	MIRROR	5 and later
DBLSPACE	6	MODE	1 and later
DECOMP	6	MORE	2 and later
DEFRAG	6	MOVE	6
DELTREE	6	MSAV	6
DISKCOMP	1 and later	MSBACKUP	6
DISKCOPY	1 and later	MSD	6
DOSKEY	5 and later	NLSFUNC	3.3 and later
DOSSHELL	4 and later	POWER	6
EDIT	5 and later	PRINT	2 and later
EDLIN	1 and later	RECOVER	2 and later
EXE2BIN	1.1 to 3.3, 5 and later	REPLACE	3.2 and later
EXPAND	5 and later	RESTORE	2 and later
FASTHELP	6	SELECT	3 and later
FASTOPEN	3.3 and later	SETVER	5 and later
FC	MS-DOS 2 and later	SHARE	3 and later
FDISK	2 and later	SMARTDRV	6
FIND	2 and later	SORT	2 and later
FORMAT	1 and later	SUBST	3.1 and later
GRAFTABL	3 to 5	SYS	1 and later
GRAPHICS	2 and later	TREE	2 and later
INTERLNK	6	UNDELETE	5 and later
INTERSVR	6	UNFORMAT	5 and later
JOIN	3.1 to 5	VSAFE	6
		XCOPY	3.2 and later

Table 3-2. *DOS External Commands*

Command	DOS Version	Command	DOS Version
BREAK	2 and later	IF	2 and later
CALL	3.3 and later	MKDIR (MD)	2 and later
CHCP	3.3 and later	PATH	2 and later
CHDIR (CD)	2 and later	PAUSE	1 and later
CLS	2 and later	PROMPT	2 and later
COPY	1 and later	REM	1 and later
CTTY	2 and later	REN	1 and later
DATE	1 and later	RMDIR (RD)	2 and later
DEL	1 and later	SET	2 and later
DIR	1 and later	SHIFT	2 and later
ECHO	2 and later	TIME	1 and later
ERASE	1 and later	TYPE	1 and later
EXIT	2 and later	VER	2 and later
FOR	2 and later	VERIFY	2 and later
GOTO	2 and later	VOL	2 and later

Table 3-3. *DOS Internal Commands*

Get into the habit of reading and analyzing the error messages DOS displays. In this case, the words "Bad command" inform you that you have entered an invalid command name or the name of an external command DOS could not locate. The Command Reference (that follows Chapter 45) of this book examines the DOS error messages in detail and presents steps you should follow to resolve them.

Type **CKS** again and watch your disk's activation light. Each time you type in a command name, DOS first checks to see if the command name matches one of its internal commands, which reside in memory. If so, DOS executes the command. If not, DOS searches the disk for a matching external command. When you type **CKS** and press ENTER, you should see your disk's activation light illuminate briefly as DOS searches the disk for the command.

Online Help in DOS 5

DOS, in version 5, provides a brief explanation about a command if you invoke that command using the /? switch. For example, the following entry will cause information about the DATE command to be displayed:

```
C:\> DATE  /? <ENTER>
Displays or sets the date.

DATE [ date ]

Type DATE without parameters to display the current date set-
ting and a prompt for a new one. Press ENTER to keep the same
date.
```

If you type HELP at the DOS prompt, the HELP command will display each DOS command with a one-line summary. HELP also lets you display help text for individual commands (see Chapter 43). This information is identical to that provided by the \? switch. For more detailed online help on all versions of DOS, refer to the disk offer at the beginning of this book.

Online Help for DOS 6 Commands

If you are using DOS 6, you can display help text for a specific command using the /? switch. For example, the following command would display help text about the DOS 6 DEFRAG command:

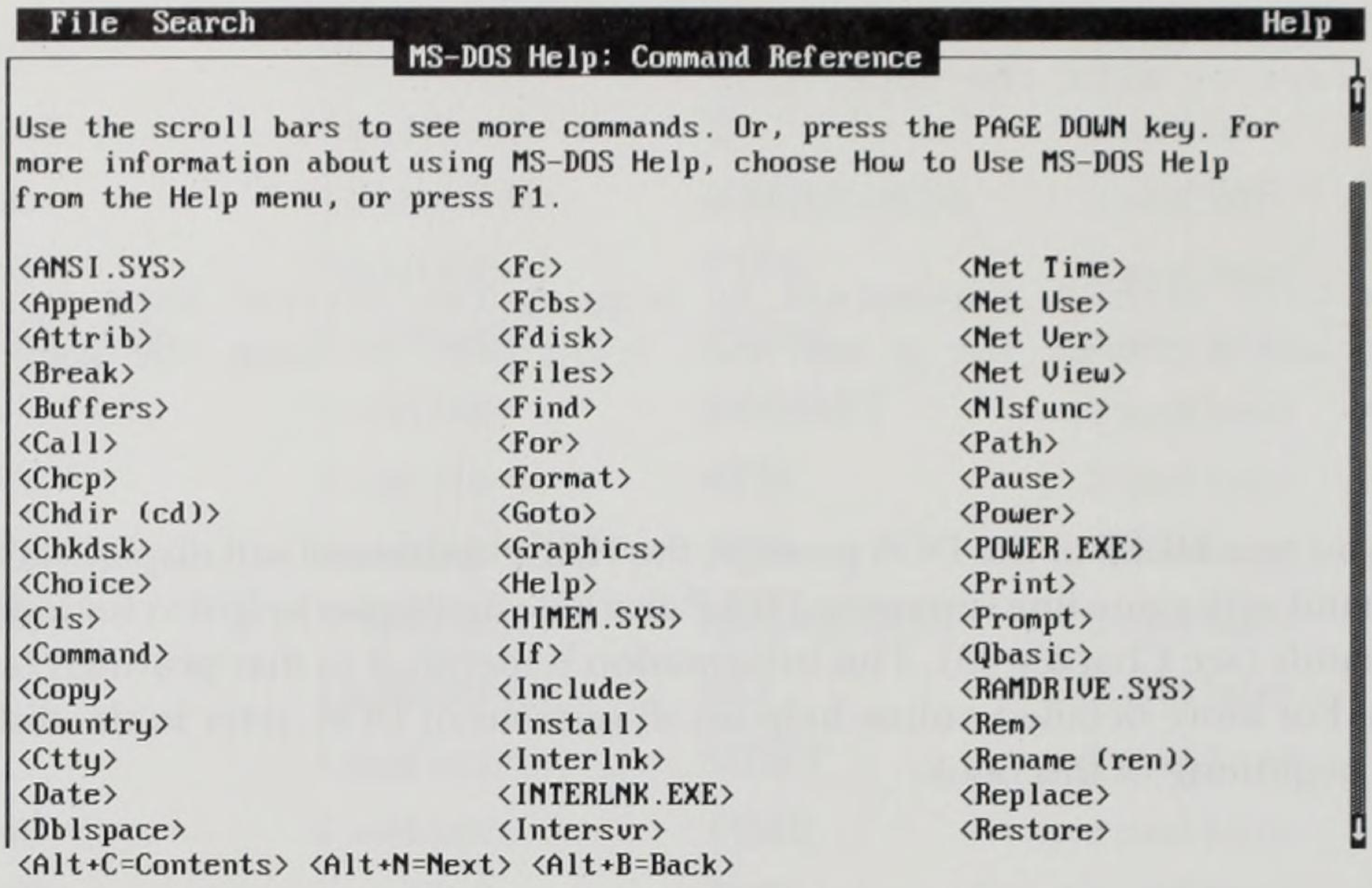
```
C:\> DEFRAG /? <ENTER>
```

DOS 6 provides two help commands. The command FASTHELP provides the same functionality as the DOS 5 HELP command just discussed. The DOS 6 HELP command invokes a menu-driven help facility, as shown in Figure 3-4.

Using your keyboard arrow, TAB keys, or mouse, you can quickly select a specific command. For example, Figure 3-5 illustrates HELP's text on the DOS 6 VSAFE command.

Using HELP's File menu, you can print the current help text or you can exit HELP back to the DOS prompt. To activate the File menu, press the ALT-F keyboard combination. Using HELP's Search command, you can quickly look up help text on a specific topic.

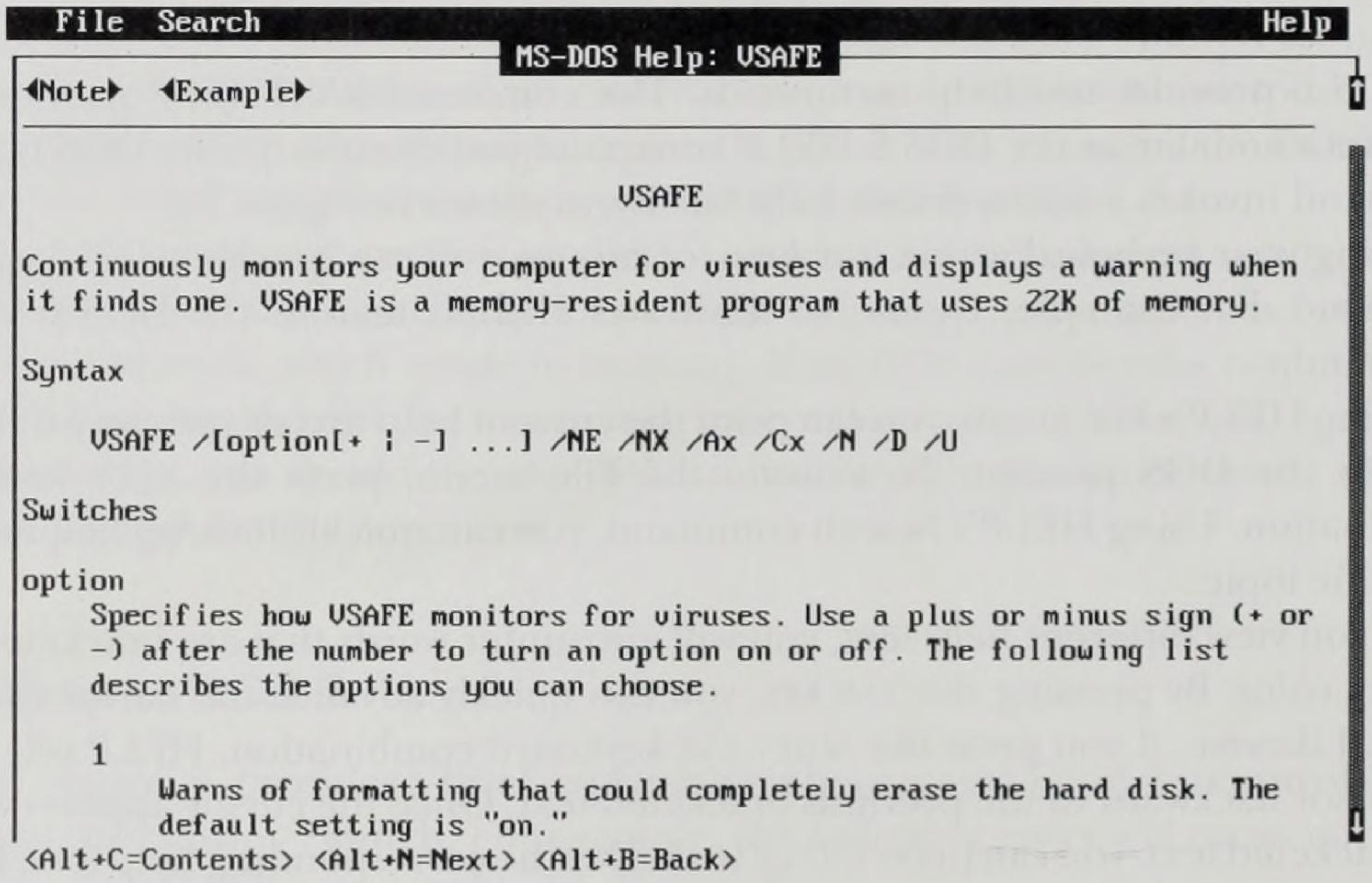
As you view different help text, you will encounter words that are bracketed in a distinct color. By pressing the TAB key, you can quickly advance the cursor to these words. Likewise, if you press the SHIFT-TAB keyboard combination, HELP will move the cursor backward to the previous bracketed text. Once the cursor appears within the bracketed text, you can press ENTER to display the corresponding help text. HELP assigns each DOS command three screens of help text. Table 3-4 briefly describes the contents of each help screen.



```
Use the scroll bars to see more commands. Or, press the PAGE DOWN key. For
more information about using MS-DOS Help, choose How to Use MS-DOS Help
from the Help menu, or press F1.

<ANSI.SYS>                  <Fc>                        <Net Time>
<Append>                    <Fcbs>                      <Net Use>
<Attrib>                    <Fdisk>                     <Net Ver>
<Break>                     <Files>                     <Net View>
<Buffers>                   <Find>                      <Nlsfunc>
<Call>                      <For>                       <Path>
<Chcp>                      <Format>                    <Pause>
<Chdir (cd)>                <Goto>                      <Power>
<Chkdsk>                    <Graphics>                  <POWER.EXE>
<Choice>                    <Help>                      <Print>
<Cls>                       <HIMEM.SYS>                 <Prompt>
<Command>                   <If>                        <Qbasic>
<Copy>                      <Include>                   <RAMDRIVE.SYS>
<Country>                   <Install>                   <Rem>
<Ctty>                      <Interlnk>                  <Rename (ren)>
<Date>                      <INTERLNK.EXE>              <Replace>
<Dblspace>                  <Intersvr>                  <Restore>
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
```

Figure 3-4. The DOS 6 HELP main menu

Finally, HELP provides three ALT-key combinations: ALT-C displays the HELP table of contents (or command list), ALT-N directs HELP to advance one screen, and ALT-B directs HELP to back up one screen.



```
◄Note► ◄Example►

                               VSAFE

Continuously monitors your computer for viruses and displays a warning when
it finds one. VSAFE is a memory-resident program that uses 22K of memory.

Syntax

    VSAFE /[option[+ | -] ...] /NE /NX /Ax /Cx /N /D /U

Switches

option
    Specifies how VSAFE monitors for viruses. Use a plus or minus sign (+ or
    -) after the number to turn an option on or off. The following list
    describes the options you can choose.

    1
        Warns of formatting that could completely erase the hard disk. The
        default setting is "on."
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
```

Figure 3-5. Online help for the DOS 6 VSAFE command

Screen Title	Contents
Notes	A discussion of the command and its characteristics
Examples	Sample commands
Syntax	Illustrates the command's format and syntax, explaining the various command-line arguments

Table 3-4. The Contents of each DOS 6 HELP Screen

SUMMARY

CLS

CLS is an abbreviation for clear screen. CLS erases your screen's current contents. When CLS completes, DOS redisplays its prompt in the upper-left corner of your screen. To invoke the CLS command, type CLS at the DOS prompt and press ENTER:

```
C:\> CLS <ENTER>
```

VER

VER is an abbreviation for version. The VER command displays your DOS version number on the screen. DOS version numbers consist of two parts, a major and minor version number. In "DOS 5.0," for example, 5 is the major version number and 0 is the minor number. When software developers release a new version of a program, the developers increment the major version number if a large number of changes to the program have been made. In the case of DOS, version 4 becomes version 5. If instead, the changes are only small updates or error corrections, the developers increment the minor version number. Using DOS once again, version 3.2 becomes version 3.3. To display your current version number, invoke theVER command, as shown here:

```
C:\> VER <ENTER>
```

SUMMARY

TIME

The TIME command lets you set or display the current system time. To set the system time, you must specify the hours and minutes of the desired time in the form *HH:MM.* If you are using DOS 3.3 or prior, you must use 24-hour military time. If you are using DOS 4 or later, you can append the letter A or P to your time to indicate A.M. or P.M. The simplest way to invoke TIME is shown here:

```
C> TIME <ENTER>
```

DATE

The DATE command lets you set or display your computer's system date. To view the date without changing it, press ENTER in response to DATE's prompt for a new date. If you want to change the date, type in the month, day, and year in the form *mm-dd-yy.* The simplest way to invoke DATE is

```
C> DATE <ENTER>
```

Hands On

To begin, press ENTER several times.

```
C> <ENTER>
C> <ENTER>
C> <ENTER>
```

Next, use CLS to clear your screen display.

```
C> CLS <ENTER>
```

Use VER to display the current DOS version number.

```
C> VER <ENTER>
```

Repeat VER, but this time, misspell it as "VAR" without pressing ENTER.

```
C> VAR_
```

Use BACKSPACE to erase the letters R and A and correct the command.

Next, use TIME to display the system time.

```
C> TIME <ENTER>
```

If your system time is incorrect, set the correct time. Repeat the process with DATE.

```
C> DATE <ENTER>
```

Review

1. How does an internal command differ from an external command?
2. What is the command line?
3. How do the following commands differ?

```
C> DATE <ENTER>

C> date <ENTER>
```

4. How do you determine the current DOS version?
5. What is the DATE command to set the system date to 12/25/93?

Answers

1. To increase your system performance, DOS provides two kinds of commands, internal commands and external commands. Internal commands include the

smaller and more commonly used DOS commands. Internal commands are so named because DOS keeps the commands in your computer's fast electronic memory whenever DOS is active. Because internal commands are stored in memory, DOS can execute them very quickly. External commands, on the other hand, reside on your computer's disk. External commands are larger, less commonly used commands. When you execute an external command, DOS must locate the command on disk and then load the command into memory before executing it. All applications, such as your word processor or spreadsheet, are made up of external commands.

2. The command line is the line of information you type at the DOS prompt, before pressing ENTER, to execute a command. The command line may simply contain a command name, or it may contain additional information such as a disk drive letter, filename, or system date.
3. The commands are the same. DOS is not sensitive to the case of the letters.
4. Type the command **VER** at the prompt to display the current DOS version number.
5. You can set the system date two ways. First, simply invoke DATE and type in the date, as shown here.

```
C:\> DATE <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): 12-25-93 <ENTER>
```

Second, you can include the date in DATE's command line:

```
C:\> DATE  12-25-93 <ENTER>
```

Advanced Concepts

External DOS commands have either the EXE or COM extension. You may wonder why DOS just doesn't use one or the other. As it turns out, in addition to having different extensions, EXE (executable) and COM (command) files have different internal formats and characteristics. DOS uses COM files for smaller commands (less than 64Kb). In addition, DOS has less freedom over where and how it loads COM files into memory. Because of these size and placement restrictions, however, DOS is able to load COM programs into memory faster than it can EXE programs. Unfortunately, the choice of whether a program becomes a COM or EXE file is made by the programmer when he or she creates the program. You can generally consider that a COM extension indicates a single command, while an EXE extension indicates an entire executable program.

Chapter 15 discusses DOS batch files, in which several command calls can be lumped together. One of the rules you should follow when creating DOS batch files is never create a batch file with the same name as an existing DOS command. Assume, for example, that you create a batch file named CLS.BAT. When you type **CLS** to execute the batch file, DOS will instead execute its internal CLS command. Remember, DOS always searches its internal commands first for a matching command, then looks for an external command on disk.

If you assign the name of an external command to a batch file, such as FORMAT.BAT, the rule for determining whether DOS will execute the batch file or external command depends on which file DOS encounters first, FORMAT.COM or FORMAT.BAT. Avoid such confusion—don't use DOS command names for names of batch files.

Key Terms

Internal command A command that resides in memory whenever DOS is active. Examples of internal commands include CLS, DATE, and TIME. Because internal commands reside in memory, DOS can execute them very quickly.

External command A command that resides on disk. Before an external command can run, DOS must first locate the command on disk and load it into memory. The majority of DOS commands are external commands. All applications, such as your word processor or spreadsheet, are made up of external commands.

Command line To execute a DOS command, you type the command's name at the DOS prompt and press ENTER. The information you type at the prompt before you press ENTER is your command line. Many DOS commands let you specify additional information, such as a filename, in the command line.

Chapter 4

Getting Started with DOS Files

To save and retrieve information from one computer session to the next, you must store information on disk. DOS organizes the information you store on disk by letting you group related pieces of information into *files.* As you create letters, reports, and spreadsheets, you will store each in its own file, much like you would place related information into a paper file in an office. Each file you create must have a unique name. You should choose meaningful names that describe the file's contents. In this chapter you will learn how to list the names of each file on your disk, as well as each file's size and the date and time the file was created. You will also examine several of the key DOS filenames as well as rules you should follow when you name your own files.

A Directory Is a List of Related Files

As you work with different programs, such as a word processor, spreadsheet, or database, you will create files to store your information. To help you organize your files, DOS lets you separate the files into related lists called *directories.* In most cases, users will have a directory each for their word processing files, DOS command files, and any other application programs they are using. The best way to view directories on your disk is as drawers of a filing cabinet into which you place related files, as shown in 4-1.

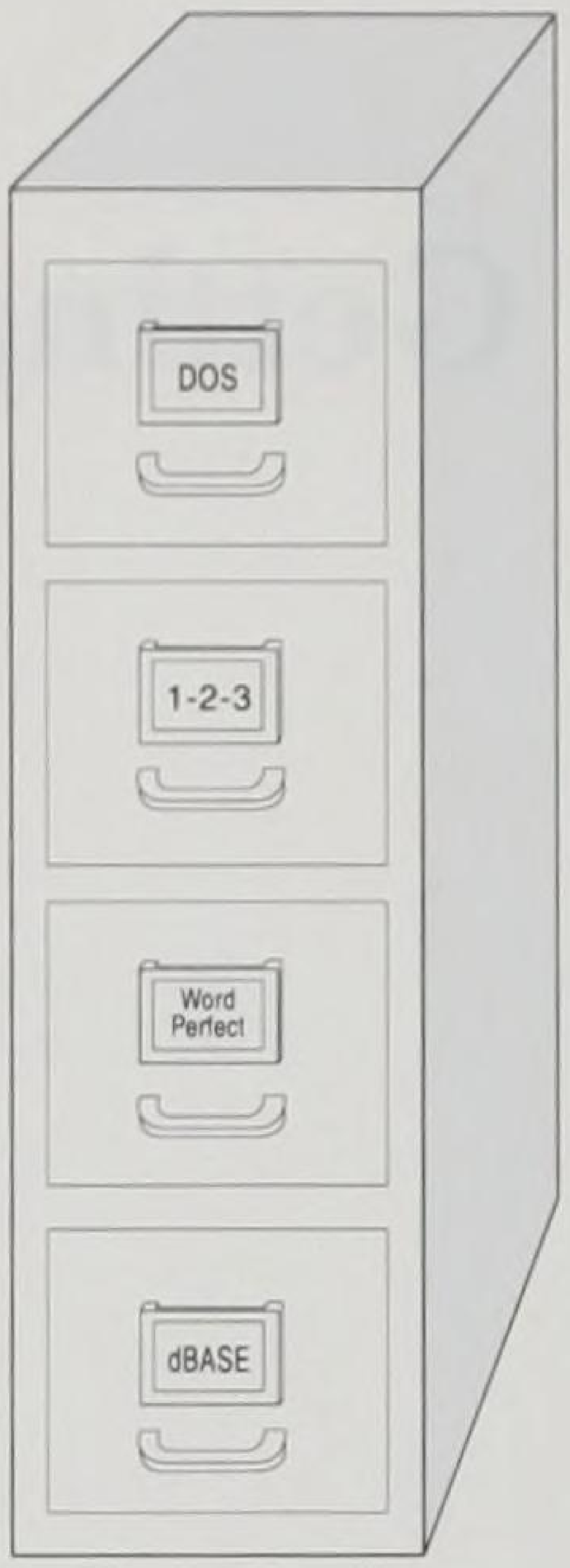


Figure 4-1. *DOS lets you organize files using directories as you would use the drawers of a filing cabinet*

Unlike a filing cabinet, which has a fixed number of drawers, you can create a virtually unlimited number of directories on your disk. In Chapter 10 you will learn how to create, select, and remove directories on your disk using the MKDIR, CHDIR, and RMDIR commands. For now, however, simply understand that a directory is a list of related files.

Using the DIR Command

As you've just read, a directory is a list of related files. Depending on the different programs installed on your disk, your disk may already contain several directories. The DIR command lets you display a list of the files in a directory. DIR is an abbreviation for directory. The simplest way to invoke DIR is to type **DIR** and press ENTER, as shown here:

```
C:\> DIR <ENTER>
```

DIR will display a list of filenames on your screen. This list only contains files in the current directory. As you will learn in Chapter 10, the current directory is conceptually similar to the filing cabinet drawer that is currently open. The directory listing will display filenames and possibly names of other directories.

DOS *filenames* each consist of two parts, an eight-character *base name* and a three-character *extension*. The DIR command displays each file's base name, extension, size, and the date and time the file was created or last changed. Figure 4-2 illustrates a typical directory entry.

As you read in Chapter 1, DOS refers to file and disk sizes in bytes. A byte, as you will recall, is a character of information.

Some of the entries DIR displays may use "<DIR>". DOS uses this designation to signify a directory. At a minimum, your hard disk should have a directory named DOS that contains the external DOS commands.

Understanding Filenames and Extensions

Each file in a directory must have a unique name. DOS filenames can use up to eight characters for the base name and up to three characters for the extension. The filename should be meaningful so it can help you determine the file's contents. The eight-character base name typically describes specific information about the file. The three-character extension specifies the file's general category. For example, you could assign the extension .LTR for files containing letters. Likewise, for reports, you might use RPT. Table 4-1 describes several commonly used file extensions.

In addition to the common file extensions listed in Table 4-1, DOS uses several file extensions to describe the different operating system files. Table 4-2 lists different DOS file extensions you may encounter.

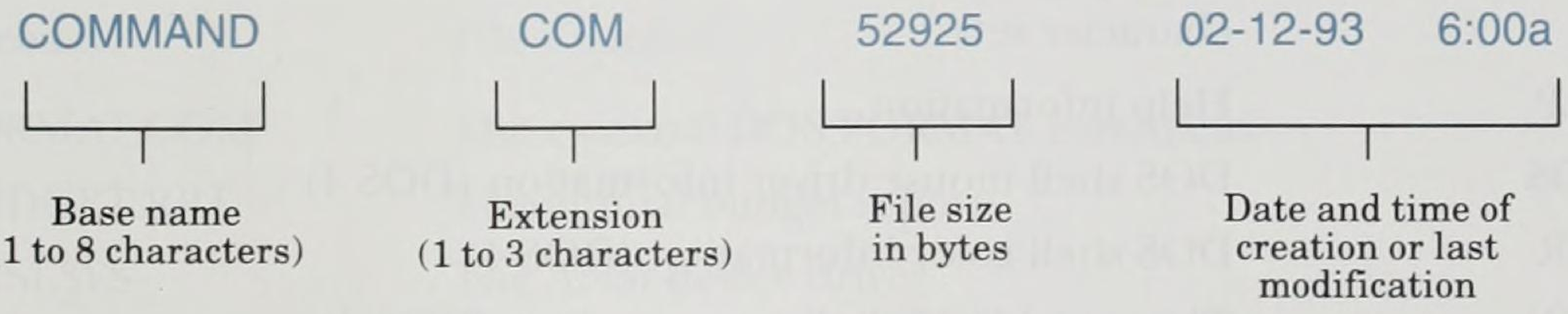

COMMAND
COM
52925
02-12-93
6:00a
Base name (1 to 8 characters)
Extension (1 to 3 characters)
File size in bytes
Date and time of creation or last modification

Figure 4-2. DIR displays each file's base name, extension, size, and date and time stamps

File Extension	File Category
LTR	Letter
MEM	Memo
RPT	Report
DAT	Numeric data
BAS	BASIC program
TXT	Word processing text document

Table 4-1. *Common User File Extensions*

When you specify a DOS filename, you must separate the base filename and extension using a period, as shown in Figure 4-3.

By using all eight of the base name characters, you can produce meaningful filenames that describe the file's contents. Table 4-3 illustrates several meaningful filenames that describe each file's contents.

File Extension	File Category
COM	DOS command
EXE	Executable file
SYS	DOS system file typically containing a device driver
BAT	DOS batch file containing a series of DOS commands
CPI	Code-page information file used to define international character sets
HLP	Help information
MOS	DOS shell mouse driver information (DOS 4)
CLR	DOS shell color information (DOS 4)
MEN	The initial DOS shell menu options (DOS 4)
PRO	Printer profiles for the GRAPHICS command
INI	DOS shell or Windows initialization values

Table 4-2. *File Extensions Commonly Used by DOS*

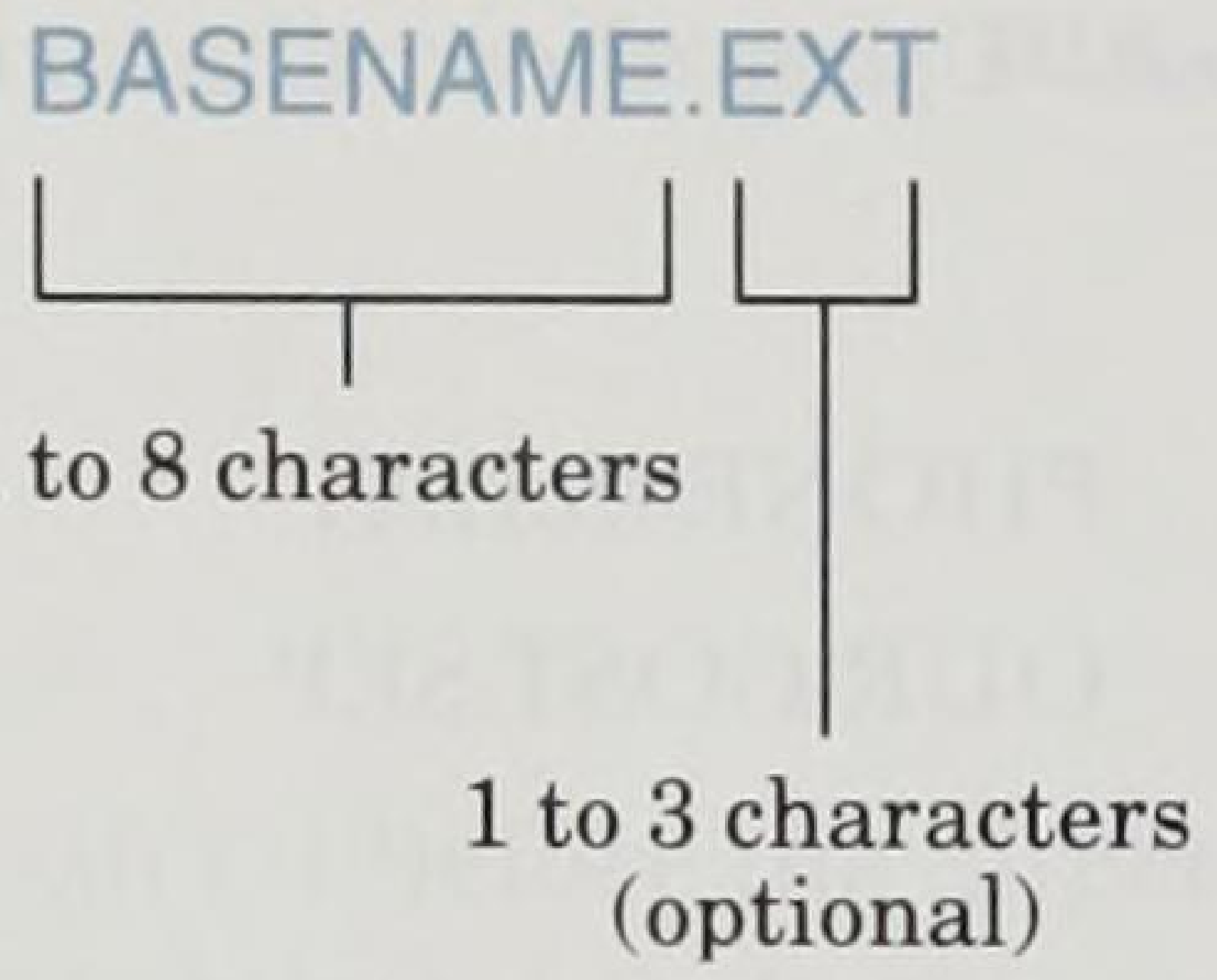


Figure 4-3. *Separate a file's base name and extension with a period*

Do not overlook the importance of assigning meaningful names to your files. Although it is easy to use a filename such as A or B, such names certainly won't help you later recall the information the file contains.

Valid Filename Characters

A file's base name can contain from one to eight characters—the three-character extension is optional. DOS lets you use the letters A through Z and the numbers 0 through 9 in your filenames. In addition, your filenames can include one or more of the following characters.

~ ! @ # $ ^ & ( ) - _ { } '

Filename	File Contents
FORMAT.COM	The external DOS FORMAT command
BUDGET.RPT	The annual budget report
ANSI.SYS	The ANSI device driver
IRS.LTR	A letter to the IRS
CHAPTER4.TXT	Chapter 4's word processing text

Table 4-3. *Meaningful Filenames and Extensions*

You can type in upper- or lowercase letters—for consistency, DOS always converts lowercase letters to uppercase. Using these characters, you can create filenames similar to the following:

1993.TAX
MAY&JUNE.RPT
PHONE#S.DAT
OUR-COST.SEP
MAYSALES.$$$
PROFITS!.DAT

Understanding Date and Time Stamps

In addition to the filename and size, DIR displays the date and time each file was created or last changed. You can use a file's *date and time stamps*, assigned to each file, to determine if you have the latest copy of a file, or to help you locate files you created or changed on a specific date. In Chapter 3, you learned how to set the system date and time using the DATE and TIME commands. DOS uses the system date and time when it assigns or updates a file's date and time stamps. If the system date or time is incorrect, so too will be the file's date and time stamps.

The only time DOS changes a file's date and time stamps is when the file's contents actually change. If a program views a file's contents, for example, the file's contents don't change, so DOS will not change the date and time stamps.

Determining the Number of Files and Available Disk Space

After DIR displays the entries in a directory, DIR displays a count of the number of files it displayed, the amount of disk space the files consume, and the amount of available disk space, as shown here:

```
90 file(s)  2236185 bytes
           16395328 bytes free
```

Many of the software packages you will purchase specify the minimum amount of free space your disk must contain before you can successfully install the file. Using the DIR command, you can quickly determine the amount of available disk space on your computer.

Listing Files in a Specific Directory

You have read that a directory is a list of files. In Chapter 10 you will learn how to quickly move from one directory to another, as you would open and close drawers of a filing cabinet. For now, however, you can use the DIR command to list the files in your DOS directory. To do so, issue the following DIR command:

```
C:\> DIR \DOS <ENTER>
```

You must include the backslash (\) before the directory name. Make sure you use the backslash (\) and not a forward slash (/). In this case, DIR will list the files in the DOS subdirectory, as shown here:

```
C:\> DIR \DOS <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

.            <DIR>     11-23-92   9:26p
..           <DIR>     11-23-92   9:26p
DBLSPACE BIN     50284 02-12-93   6:00a
FORMAT   COM     22717 02-12-93   6:00a
NLSFUNC  EXE      7036 02-12-93   6:00a
COUNTRY  SYS     17066 02-12-93   6:00a
KEYB     COM     14983 02-12-93   6:00a
KEYBOARD SYS     34694 02-12-93   6:00a
ANSI     SYS      9065 02-12-93   6:00a
ATTRIB   EXE     11165 02-12-93   6:00a
CHKDSK   EXE     12908 02-12-93   6:00a
EDIT     COM       413 02-12-93   6:00a
:        :         :     :         :
:        :         :     :         :
MSAVHELP OVL     29828 02-12-93   6:00a
MSAVIRUS LST     35520 02-12-93   6:00a
VSAFE    COM     62576 02-12-93   6:00a
MWAVDOSL DLL     44736 02-12-93   6:00a
MWAVDRVL DLL      7744 02-12-93   6:00a
MWAVDLG  DLL     36368 02-12-93   6:00a
MWAVSCAN DLL    151568 02-12-93   6:00a
MWAV     EXE    142640 02-12-93   6:00a
MWAVABSI DLL     54576 02-12-93   6:00a
```

```
MWAV      HLP      25663 02-12-93   6:00a
DEFAULT   SET       4482 02-08-93   7:51p
DEFAULT   SLT         21 02-08-93   1:58p
MWAVSOS   DLL       7888 02-12-93   6:00a
MWAVMGR   DLL      21712 02-12-93   6:00a
MWAVTSR   EXE      17328 02-12-93   6:00a
COMMAND   COM      52925 02-12-93   6:00a
QBASIC    INI        132 02-20-93  12:10p
MOUSE     INI         28 02-03-93  11:54a
DEFAULT   CAT         66 02-08-93   7:51p
       128 file(s)      5826105 bytes
                       16762880 bytes free
```

Depending on your DOS version, the number of files the directory contains, as well as the actual filenames, may differ. Listing the contents of the DOS directory in this way is conceptually similar to opening the filing cabinet drawer labeled "DOS" and listing the files the drawer contains, as shown in Figure 4-4.

If the file entries scrolled past you on the screen faster than you could view them, include the /P *switch* (a switch is a letter or letters, typically preceded by a forward slash (/), that further refines a command) in the DIR command line, as follows.

```
C:\> DIR \DOS /P <ENTER>
```

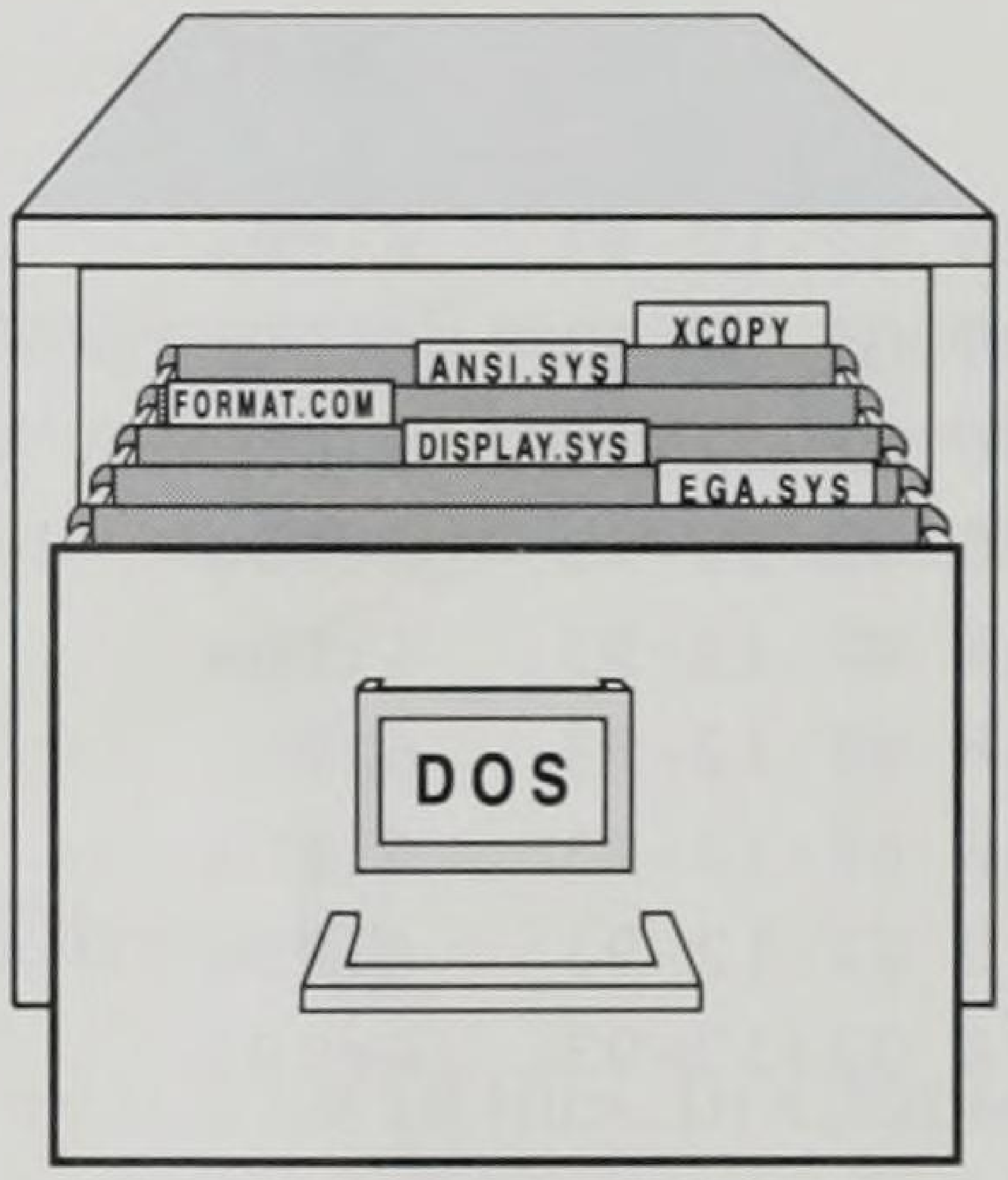


Figure 4-4. *Listing the files in the DOS directory*

The /P switch directs DIR to temporarily suspend its output with each screenful of files until you press a key to continue. When you use the /P switch, DIR will prompt you with a message to press a key to continue, as shown here:

```
C:\> DIR \DOS /P <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

.            <DIR>     11-23-92   9:26p
..           <DIR>     11-23-92   9:26p
DBLSPACE BIN     50284 02-12-93   6:00a
FORMAT   COM     22717 02-12-93   6:00a
NLSFUNC  EXE      7036 02-12-93   6:00a
COUNTRY  SYS     17066 02-12-93   6:00a
KEYB     COM     14983 02-12-93   6:00a
KEYBOARD SYS     34694 02-12-93   6:00a
ANSI     SYS      9065 02-12-93   6:00a
ATTRIB   EXE     11165 02-12-93   6:00a
CHKDSK   EXE     12908 02-12-93   6:00a
EDIT     COM       413 02-12-93   6:00a
EXPAND   EXE     16129 02-12-93   6:00a
MORE     COM      2546 02-12-93   6:00a
MSD      EXE    158470 02-12-93   6:00a
QBASIC   EXE    194309 02-12-93   6:00a
RESTORE  EXE     38294 02-12-93   6:00a
SYS      COM      9379 02-12-93   6:00a
UNFORMAT COM     12738 02-12-93   6:00a
Press any key to continue . . .
```

Take a close look at the filenames in the DOS directory listing. Many of the files have the EXE or COM extension. Such files contain external DOS commands. As you read in Chapter 3, DOS commands are either *internal* commands, which reside in your computer's fast electronic memory, or *external* commands, which reside on disk. When you execute an external command, DOS must first load the command from disk into memory before the command can execute.

Displaying the Directory Listing for a Specific File

In the previous section you learned how to display a list of files in a specific directory by including the directory's name, as shown in the next examples.

```
C:\> DIR \DOS <ENTER>
```

In this case, DIR will display the directory entry for every file in the DOS directory. If you want to display the directory entry for a specific file, include the filename in the directory command. For example, the following DIR command displays the directory listing for the file FORMAT.COM, which resides in the DOS directory.

```
C:\> DIR \DOS\FORMAT.COM <ENTER>
```

In this case, DIR displays the following:

```
C:\> DIR \DOS\FORMAT.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

FORMAT   COM    22717 02-12-93   6:00a
        1 file(s)    22717 bytes
                  16762880 bytes free
```

Conceptually, this DIR command is equivalent to opening the filing cabinet drawer labeled "DOS" and accessing the file labeled FORMAT.COM, as shown in Figure 4-5.

As you create more files on your disk, you will frequently use DIR in this manner to locate specific files.

Displaying a Wide Directory Listing

As discussed, the DIR command displays file base names, extensions, sizes, and the date and time stamps. If you only want filenames and extensions, you can use the /W switch to direct DIR to display a wide directory listing with five filenames across the screen. The following command uses /W to display the files in the DOS directory.

```
C:\> DIR \DOS /W <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

[.]          [..]          DBLSPACE.BIN  FORMAT.COM    NLSFUNC.EXE
COUNTRY.SYS  KEYB.COM      KEYBOARD.SYS  ANSI.SYS      ATTRIB.EXE
CHKDSK.EXE   EDIT.COM      EXPAND.EXE    MORE.COM      MSD.EXE
QBASIC.EXE   RESTORE.EXE   SYS.COM       UNFORMAT.COM  README.TXT
DEBUG.EXE    FDISK.EXE     DOSSHELL.VID  DOSSHELL.INI  DOSSHELL.GRB
  :             :              :             :             :
  :             :              :             :             :
MWAVSCAN.DLL MWAV.EXE      MWAVABSI.DLL  MWAV.HLP      DEFAULT.SET
DEFAULT.SLT  MWAVSOS.DLL   MWAVMGR.DLL   MWAVTSR.EXE   COMMAND.COM
QBASIC.INI   MOUSE.INI     DEFAULT.CAT
    128 file(s)    5826105 bytes
            16762880 bytes free
```

As you can see, using /W, you direct DIR to only display filenames and extensions, suppressing the display of each file's size and date and time stamp.

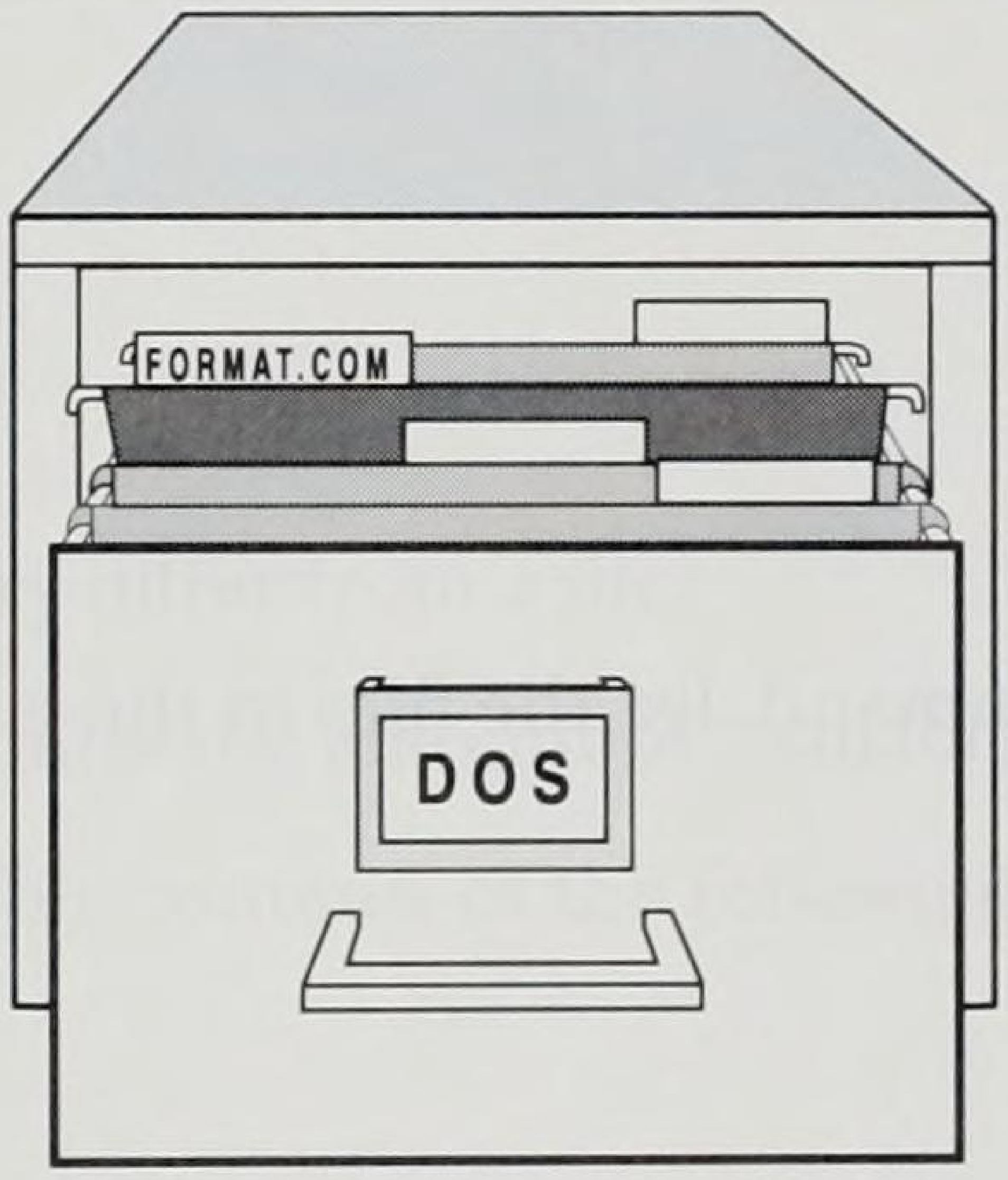


Figure 4-5. Accessing the FORMAT.COM file

SUMMARY

Directories

To help you organize your files on disk, DOS lets you group files into lists called directories. DOS directories are conceptually similar to drawers of a filing cabinet, which also contain related files.

Filenames

DOS filenames have two parts, an eight-character base name and a three-character extension. When you specify a filename, you separate the base name and extension using a period. Your filenames should be as meaningful as possible to help you understand the file's contents simply by examining the filename. The eight-character base name should provide specifics about the file, whereas the three-character extension should specify the file's type.

File Date and Time Stamps

For every file you create on your disk, DOS assigns a date and a time stamp. Each time you update a file's contents, DOS updates the file's date and time stamps using the current system date and time. Using the date and time stamps, you can determine if you have the most recent copy of a file. You can also search for files created on a specific day. If your system date and time is wrong, DOS will assign the wrong date or time to the file date and time stamp.

Hands On

Using the following DIR command, list the files in the current directory.

```
C:\> DIR <ENTER>
```

Note each file's base name, extension, size, and date and time stamp. Also note the amount of available space on your disk. As discussed, entries containing "<DIR>"

are directories containing additional lists of filenames. Note the directories that reside on your disk. The directory names should give you an indication of the different applications programs that are installed on your disk.

If the files in your directory scroll past you faster than you can read them, invoke DIR using the /P switch, as shown here:

```
C:\> DIR /P <ENTER>
```

Next, use the following DIR command to display the files in the DOS directory.

```
C:\> DIR \DOS <ENTER>
```

Next, using the /W switch, display the files in your DOS directory using a wide-directory format.

```
C:\> DIR \DOS /W <ENTER>
```

Last, use the following command to display the directory entry for the file DISKCOPY.COM, which resides in the DOS directory.

```
C:\> DIR \DOS\DISKCOPY.COM <ENTER>
```

Review

1. How does a directory differ from a file?
2. How can you determine the amount of available space on your disk?
3. What are the probable contents of the following files?

   DIRSORT.BAT
   LABEL.COM
   BUDGET93.RPT
   BUSINESS.LTR

4. What is the purpose of a file extension?
5. How many characters can a DOS filename contain?
6. What is the purpose of the DIR /W switch?

Answers

1. To organize the information you store on your disk, DOS lets you group related pieces of information into a file, much like you would use a paper file within an office. A directory, on the other hand, is a list of related files, similar to a filing cabinet drawer.
2. The last piece of information the DIR command displays is the number of files in the directory and the amount of available disk space.

```
90 file(s)    2236185 bytes
             16387136 bytes free
```

3. The probable contents of these files are as follows:

DIRSORT.BAT	A DOS batch file that sorts a directory
LABEL.COM	The LABEL command
BUDGET93.RPT	A report describing the 1993 budget
BUSINESS.LTR	A business letter

4. A DOS filename consists of two parts, an eight-character base name and a three-character extension. The extension provides information on the file's type, such as a command (COM or EXE), a letter (LTR), or a report (RPT).
5. A DOS filename can have an eight-character base name and a three-character extension. You must separate the base name and extension with a period, as shown here: BASENAME.EXT.
6. The /W switch directs DIR to suppress its display of each file's size and date and time stamps, displaying instead only base names and extensions. Because the base names and extensions don't take up much space, DIR can display five filenames per line.

Advanced Concepts

By default, the DIR command displays file entries in the same order the files appear in the directory listing. To obtain a sorted directory listing, users often *pipe* (redirect) the output of the DIR command into SORT, as shown here:

```
C:\SORTTEST> DIR | SORT <ENTER>
```

In this case, DOS will display the file entries by name in ascending order (A to Z), as shown here:

```
C:\SORTTEST> DIR | SORT <ENTER>
                167342080 bytes free
       10 file(s)   13824 bytes
 Directory of C:\SORTTEST
 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
.            <DIR>    03-03-93   8:22a
..           <DIR>    03-03-93   8:22a
AAAAAAAA AAA     5120 02-27-93   1:36p
BBBBBBBB BBB     2176 01-09-93   1:33p
CCCCCCCC CCC      640 02-22-93  11:23a
DDDDDDDD DDD     1024 11-08-93   1:22p
EEEEEEEE EEE      512 02-20-93  11:23a
FFFFFFFF FFF      896 02-04-93  10:20a
GGGGGGGG GGG      768 02-08-93   9:33a
HHHHHHHH HHH     2688 02-06-93   1:07p
```

Because of the sort order, the command displays the number of files in the directory and amount of available disk space before the filenames.

If you sort the files in descending order using SORT's /R switch, the ordering becomes worse, as shown here:

```
C:\SORTTEST> DIR | SORT /R <ENTER>
HHHHHHHH HHH   2688 02-06-93  1:07p
GGGGGGGG GGG    768 02-08-93  9:33a
FFFFFFFF FFF    896 02-04-93 10:20a
```

```
EEEEEEEE EEE     512 02-20-93 11:23a
DDDDDDDD DDD    1024 11-08-93  1:22p
CCCCCCCC CCC     640 02-22-93 11:23a
BBBBBBBB BBB    2176 01-09-93  1:33p
AIBLAEEB           0 03-03-93  8:27a
AIBLAEDM           0 03-03-93  8:27a
AAAAAAAA AAA    5120 02-27-93  1:36p
..           <DIR>    03-03-93  8:22a
.            <DIR>    03-03-93  8:22a
 Volume Serial Number is 1661-AB75
 Volume in drive C is DOS
 Directory of C:\SORTTEST
       12 file(s)    13824 bytes
                 167342080 bytes free
```

If you examine the last five lines (those not containing a directory listing), you'll find that each contains the lowercase letter e. Using the FIND command /V switch, you can remove these lines from the output, cleaning up the sorted directory listing:

```
C:\> DIR | SORT /R | FIND /V "e" <ENTER>
```

If you use this command on a regular basis, you may simply want to place it into a batch file named DESCEND.BAT.

Just as there are times when you want to sort the directory by name, there may be times when you want to sort by size. The following command uses SORT's /+*n* (where *n* is a number) column switch to display the directory listing sorted by size (beginning in column 13).

```
C:\> DIR | SORT /+13 | FIND /V "e" <ENTER>
```

If you are using DOS 5 or later, DIR provides several new command-line switches that give you tremendous flexibility.

Using the /O Switch to Specify a Sort Order

In DOS 5 or later, DIR's /O switch lets you specify the order in which you want DIR to display directory entries. The format of this switch is /O:*options*, where *options* are one or more of the entries specified in Table 4-4.

The following DIR command sorts the directory by name in reverse (Z to A) order:

```
C:\> DIR /O:-N <ENTER>
```

Likewise, this command displays the files sorted by size (smallest to largest):

```
C:\> DIR /O:S <ENTER>
```

Option	Order
C	By compression ratio, smallest to largest
–C	By compression ratio, largest to smallest
D	By date, oldest to newest
–D	By date, newest to oldest
E	By extension, A to Z
–E	By extension, Z to A
G	Group directories and display them before files
–G	Group directories and display them after files
N	By name, A to Z
–N	By name, Z to A
S	By size, smallest to largest
–S	By size, largest to smallest

Table 4-4 *DOS 5 or later DIR /O Sort Options*

DIR lets you specify several options at one time. For example, the following DIR command groups directories, displaying the directories before files, and then displays the files sorted by extension (A to Z):

```
C:\> DIR /O:GE <ENTER>
```

Experiment with combinations of the sort options, and you will agree they are very convenient.

Using the /A Switch to List Files by Attribute

By default, DIR displays all of the entries in the directory, regardless of the entry's attribute settings. Using the /A switch, the DOS 5 or later DIR command lets you restrict your directory listings to read-only files, hidden files, directories, and so on. The format of the /A switch is /A:*options,* where options are one or more of the entries specified in Table 4-5.

Option	Files Listed
A	Files with the archive attribute set
–A	Files with the archive attribute cleared
D	Directories
–D	All files except directories
H	All hidden files
–H	All files not hidden
R	All read-only files
–R	All files not read-only
S	All files with the system attribute set
–S	All files with the system attribute cleared

Table 4-5. DIR File Attribute Options

The following command uses the /A switch to display read-only files:

```
C:\> DIR /A:R <ENTER>
```

Likewise, the next command displays all hidden files in the root directory:

```
C:\> DIR /A:H \*.* <ENTER>
```

By combining options, the next command displays files that are not read-only but do require archiving:

```
C:\> DIR /A:-RA <ENTER>
```

The Rest of the DOS 5 or later DIR Switches

In addition to /A and /O, DOS 5 or later provides three additional DIR command switches. First, the /S switch directs DIR to display the directory entries for files that reside in subdirectories beneath the target directory. Using /S, the DIR command shown in the following example lists the directory entry for each file on your disk.

```
C:\> DIR \*.* /S <ENTER>
```

The /L switch directs DIR to display the filenames in lowercase letters. The following command displays the DOS directory files in this manner:

```
C:\> DIR \DOS /L <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

.            <DIR>      11-23-92   9:26p
..           <DIR>      11-23-92   9:26p
dblspace bin      50284 02-12-93   6:00a
format   com      22717 02-12-93   6:00a
nlsfunc  exe       7036 02-12-93   6:00a
```

```
country   sys      17066 02-12-93    6:00a
keyb      com      14983 02-12-93    6:00a
   :        :         :       :        :
   :        :         :       :        :
mwavsos   dll       7888 02-12-93    6:00a
mwavmgr   dll      21712 02-12-93    6:00a
mwavtsr   exe      17328 02-12-93    6:00a
command   com      52925 02-12-93    6:00a
qbasic    ini        132 02-20-93   12:10p
mouse     ini         28 02-03-93   11:54a
default   cat         66 02-08-93    7:51p
      128 file(s)     5826105 bytes
                     16762880 bytes free
```

Last, the /B switch directs DIR to display only base filenames and extensions, one file per line, suppressing the file size and date and time stamps. Again using the DOS directory, the output of DIR using the /B switch becomes the following:

```
C:\> DIR \DOS /B <ENTER>
DBLSPACE.BIN
FORMAT.COM
NLSFUNC.EXE
COUNTRY.SYS
KEYB.COM
KEYBOARD.SYS
ANSI.SYS
ATTRIB.EXE
CHKDSK.EXE
EDIT.COM
EXPAND.EXE
   :
   :
MWAVMGR.DLL
MWAVTSR.EXE
COMMAND.COM
QBASIC.INI
MOUSE.INI
DEFAULT.CAT
```

Note that the /B switch also suppresses the display of the number of files and available disk space.

DIR lets you combine all of its switches. For example, the following command displays a lowercase directory listing of your entire disk, showing only the base names and extensions.

```
C:\> DIR \*.* /S /L /B <ENTER>
```

Take some time now to experiment with the DOS 5 or later DIR command-line switches.

Key Terms

Directory A list of file entries used to organize your disk.

Base name The up-to-eight-character name that precedes the extension in a DOS filename. A file's base name should describe the file's contents.

Extension The up-to-three-character sequence following the base filename that indicates a file's type, such as a letter (LTR), report (RPT), or command (COM).

Date and time stamps The date and time DOS assigns to a file that correspond to the date and time the file was created or last changed.

Chapter 5

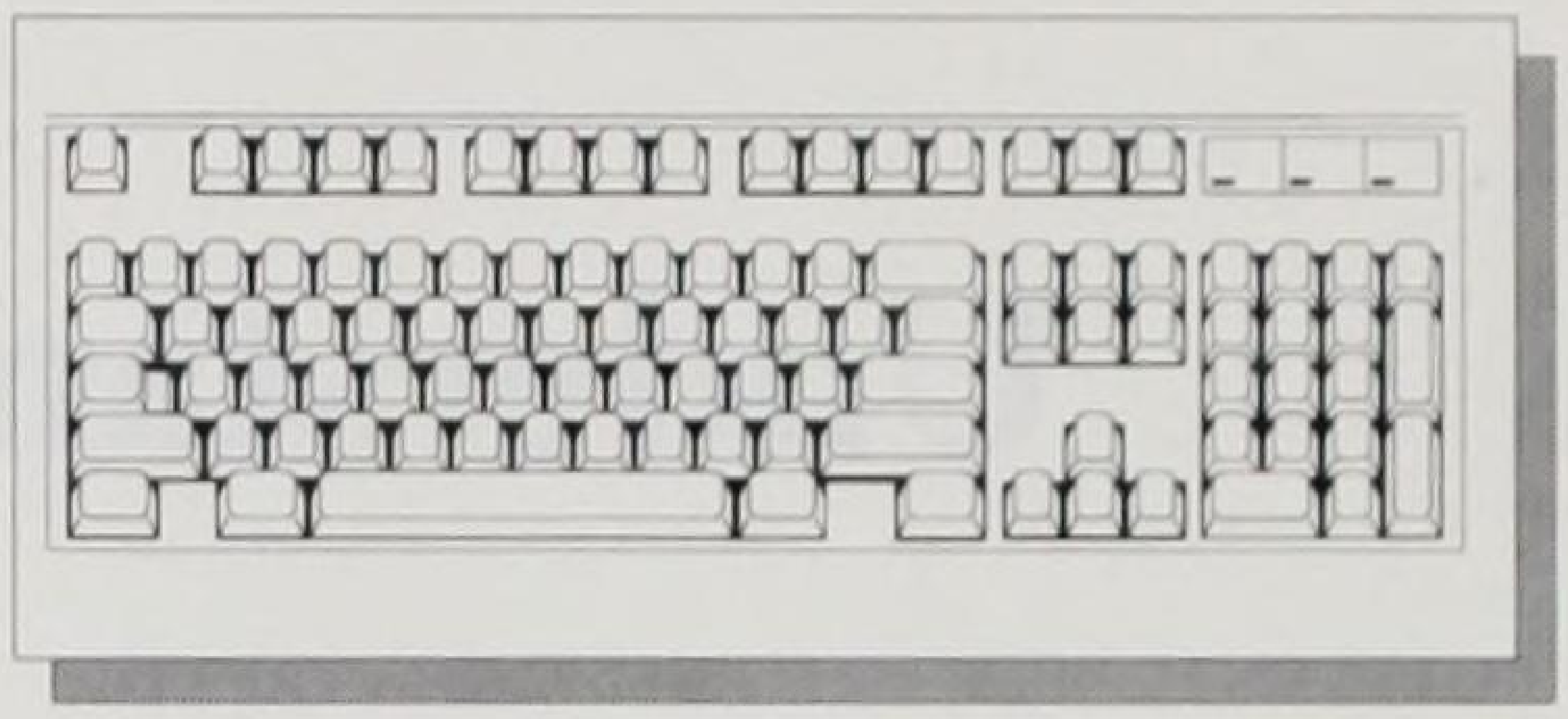
Understanding Your Keyboard

As you learned in Chapter 1, you use the keyboard to type information into your computer. The information you type might be a DOS command, a letter you are creating with your word processor, or expenses you are entering into a spreadsheet. This chapter examines your keyboard and different keyboard techniques you can use to save time and reduce keystrokes. In addition, you will learn how to use your keyboard to cancel a command and to restart DOS without having to turn your computer's power on and off.

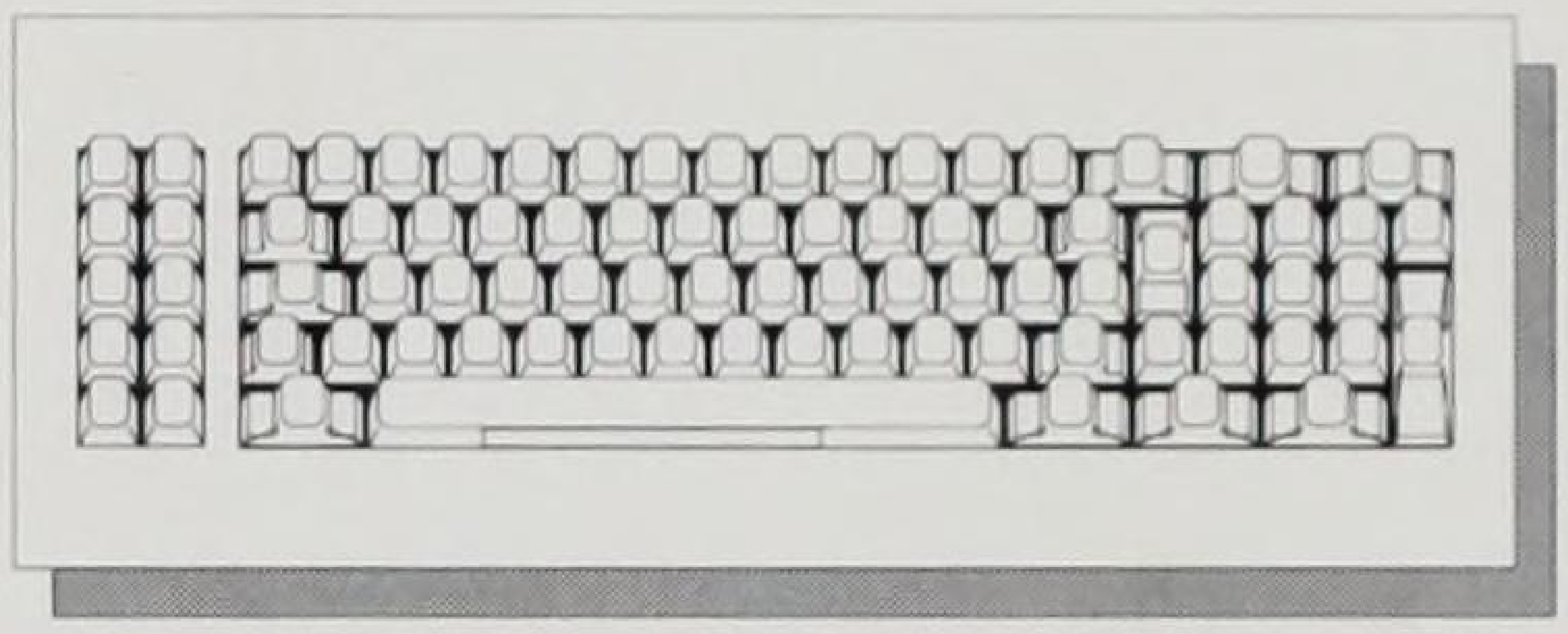
Keyboards Have Four Major Sections

Since the introduction of the IBM PC in 1981, keyboard layouts have changed to provide more capabilities and increase your ease of use. Figure 5-1 illustrates two of the most common PC keyboard layouts.

Although these keyboards may differ in key layout, each contains four primary sections: the standard typewriter keys, the function keys, the numeric keypad, and the cursor-control keys.



PC AT and PS/2 Keyboard



Original IBM PC Keyboard

Figure 5-1 *Common keyboard layouts*

Standard Typewriter Keys

The vast majority of your keyboard is laid out in the same format as a typewriter. Such key layouts are often call "QWERTY" because they are based on the layout of the upper-left row of letter keys, as shown in Figure 5-2.

Typing with a computer is much easier than typing on a typewriter because you can quickly correct mistakes using the BACKSPACE key (on some keyboards a long-stemmed left arrow directly above the ENTER key), as discussed in Chapter 3.

There are two keys that appear on your keyboard, labeled CTRL (for control) and ALT (for alternate), which don't appear on a typewriter. Note the location of these keys on your keyboard. You will use these keys in combination with other keys throughout this chapter.

Function Keys

Depending on your keyboard type, your keyboard will have keys labeled F1 through F10 (or possibly F12) called *function keys*. As you will learn, DOS, as well as many of your application programs, assigns unique functions to these keys. This section examines the three most widely used function keys in DOS—F1, F3, and F6.

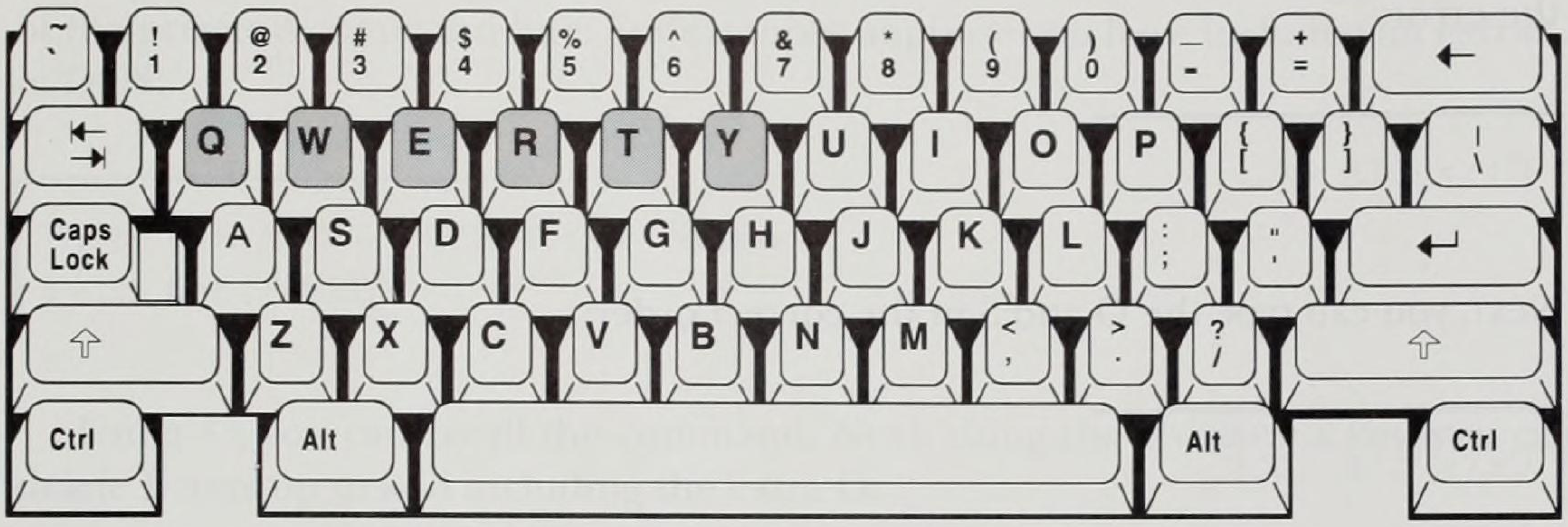

~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = ←
Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " '
Z X C V B N M < , > . ? /
Ctrl Alt Alt Ctrl

Figure 5-2 *Keyboard using the standard typewriter layout, QWERTY key positioning*

Using the Function Keys to Repeat a Command

As you work with DOS, there will be many times when you need to repeat a command or issue a slight variation of the previous command. As it turns out, each time DOS executes a command, DOS temporarily saves the command in case you want to repeat it. For example, assume you invoke the DIR command.

```
C:\> DIR <ENTER>
```

To repeat the command, you can type the command name once again, or you can instead press the F3 function key. When you press F3, DOS will immediately display the last command executed next to its prompt. To repeat the command, you can simply press ENTER. If you want to change the previous command slightly, you can. In the case of DIR, for example, you might want to add the /W switch. Likewise, if you misspell a command, you can redisplay the command using F3 and edit it using the arrow, BACKSPACE, and INS (Insert) keys.

Suppose, for example, that you have inadvertently misspelled DOS in the following command as "DSO":

```
C:\> DIR \DSO\FORMAT.COM <ENTER>
Path not found
```

Using the F3 function key, you can recall the command, as shown here:

```
C:\> DIR \DSO\FORMAT.COM_
```

Using your keyboard's BACKSPACE key, you can erase the command to the point of the error:

```
C:\> DIR \D_
```

Next, you can type the O and S in the correct order:

```
C:\> DIR \DOS_
```

Last, pressing the F3 key a second time, you can display the remainder of the command:

```
C:\> DIR \DOS\FORMAT.COM_
```

Next, instead of transposing the letters O and S, you simply forgot the O:

```
C:\> DIR \DS\FORMAT.COM <ENTER>
Path not found
```

In this case, repeat the process of pressing F3 and using the BACKSPACE key to erase the command to the point of the error as shown here:

```
C:\> DIR \D_
```

Because you need to insert a letter into the command, press the INS key. Next, type **O** and press F3. DOS will display the correct command.

```
C:\> DIR \DOS\FORMAT.COM_
```

The INS key works as a *toggle.* The first time you press the INS key, DOS enters *insert mode,* meaning DOS inserts the characters you type in front of the characters at the current position. The second time you press the INS key, DOS enters *overstrike mode,* meaning DOS will overwrite the characters at the current position. Don't let insert and overstrike modes confuse you. Many word processors use the INS key. With DOS, the only time you need to concern yourself with the INS key is when you need to insert characters into a command you are editing with the function keys.

Similarly, there may be times when you need to delete a character from the middle of the previous command line. For example, suppose you have included an extra O in DOS:

```
C:\> DIR \DOOS\FORMAT.COM <ENTER>
Path not found
```

Using F3, you can recall the command. Next, using the BACKSPACE key, you can delete letters up to and including the extra O:

```
C:\> DIR \DO_
```

Next, press the key labeled DEL (Delete). DOS will remove the current letter from the command buffer. When you press F3, DOS will append the remainder of the buffer, resulting in the correct command:

```
C:\> DIR \DOS\FORMAT.COM_
```

As you have just discovered, pressing the F3 function key lets you quickly repeat or edit the entire previous command. In some cases, however, you may find it more convenient to display the letters of the previous command one letter at a time. The F1 function key lets you do just that.

Assume, for example, that you have just issued the following DIR command:

```
C:\> DIR \DOS\FORMAT.COM
```

Next, assume you want to simply execute the DIR command for the current directory. Each time you press the F1 function key, DOS will display a single letter of the previous command. In this case, by pressing the F1 function key three times, you can display the DIR command like this:

```
C:\> DIR_
```

Depending on the amount you need to edit the previous command, you might find that using the F1 function key (or the right arrow, which serves the same purpose) is faster than using F3 in conjunction with the BACKSPACE key.

Copying Simple Files from Your Keyboard

In Chapter 4, you learned that DOS stores information on your disk in files. Using a word processor or the DOS editor, you can create your own files. In addition, if you are creating small, simple files, you can copy the file's contents directly from your keyboard. To begin, DOS assigns the device name CON (abbreviation for console) to your keyboard. Using the CON device name, the following COPY command directs DOS to copy into the file TEST.DAT input from your keyboard:

```
C:\> COPY CON TEST.DAT <ENTER>
```

When you invoke this command, DOS will advance the cursor to the start of the next line. Type the lines shown and press ENTER.

```
C:\> COPY CON TEST.DAT <ENTER>
This is the file's first line. <ENTER>
The file only contains two lines. <ENTER>
```

As before, DOS will advance the cursor and wait for you to type the file's next line. You must tell DOS you have finished entering information into the file. To do so, press the F6 function key. DOS will in turn display the characters ^Z (control Z), as shown in the following.

```
C:\> COPY CON TEST.DAT <ENTER>
This is the file's first line. <ENTER>
The file only contains two lines. <ENTER>

```

DOS uses ^Z to indicate the end of a file. You can only generate this letter by pressing the F6 key or by pressing the CTRL-Z key combination. When you press ENTER following the ^Z, DOS will create the file:

```
C:\> COPY CON TEST.DAT <ENTER>
This is the file's first line. <ENTER>
The file only contains two lines. <ENTER>
^Z <ENTER>
    1 File(s) copied

C:\>
```

Because it is commonly used to mark the end of a file you create from the keyboard, the F6 key is often called the *end-of-file key*. As you need to create small files, you may find copying the file's contents from the keyboard to be quite convenient. Beware—you can't change the contents of lines after you press ENTER. If you need to change (edit) the file's contents, use your word processor or the DOS editor.

Numeric Keypad

Every keyboard has a group of numeric keys that resemble the keys on a ten-key adding machine, as shown in Figure 5-3.

DOS itself does not make extensive use of the numeric keypad. The keypad exists to simplify your entry of numbers when you enter data into a spreadsheet or some other application that requires the entry of many numbers. Your keyboard has a key labeled NUMLOCK. Before you can use the numeric keys to generate numbers, you may have to press NUMLOCK. As you will learn next, the numeric keypad also doubles as your cursor-control keys. The NUMLOCK key works as a toggle. The first time you press NUMLOCK, DOS selects the numeric keypad's number keys. The second time you press NUMLOCK, DOS selects the cursor-control keys. Most keyboards have a small light that indicates when NUMLOCK (meaning the numeric keys) is active.

Cursor-Control Keys

Depending on your keyboard type, your keyboard may have one or two sets of arrow keys, shown in Figures 5-3 and 5-4.

Aside from the DOS editor and the DOS 5 and 6 DOSKEY command (discussed in Chapter 27), DOS doesn't make much use of the arrow keys. The arrow keys, or *cursor-control keys* are used primarily by applications to let you traverse your word processing documents or spreadsheets.

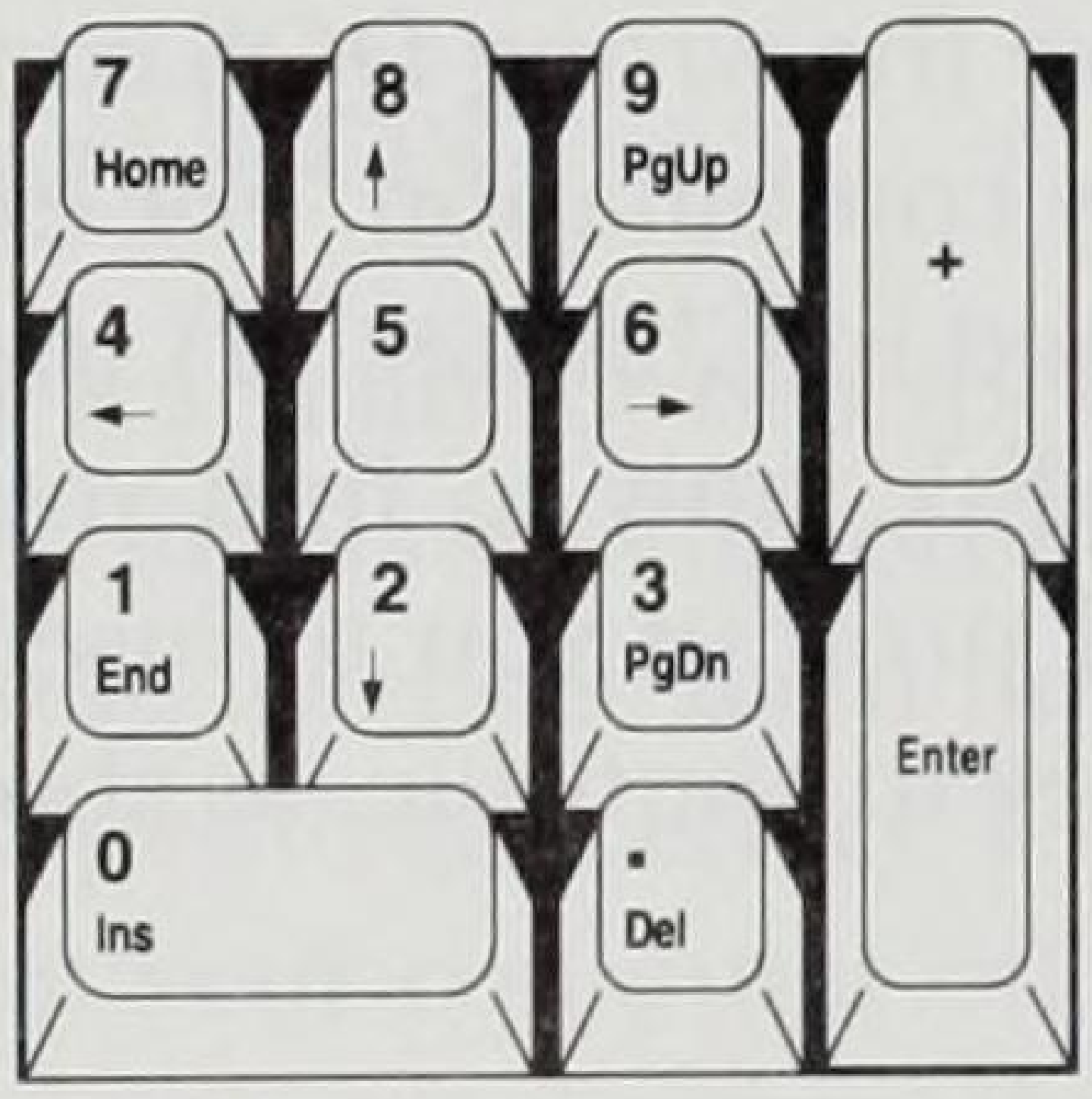


Figure 5-3. *The keyboard's numeric keypad*

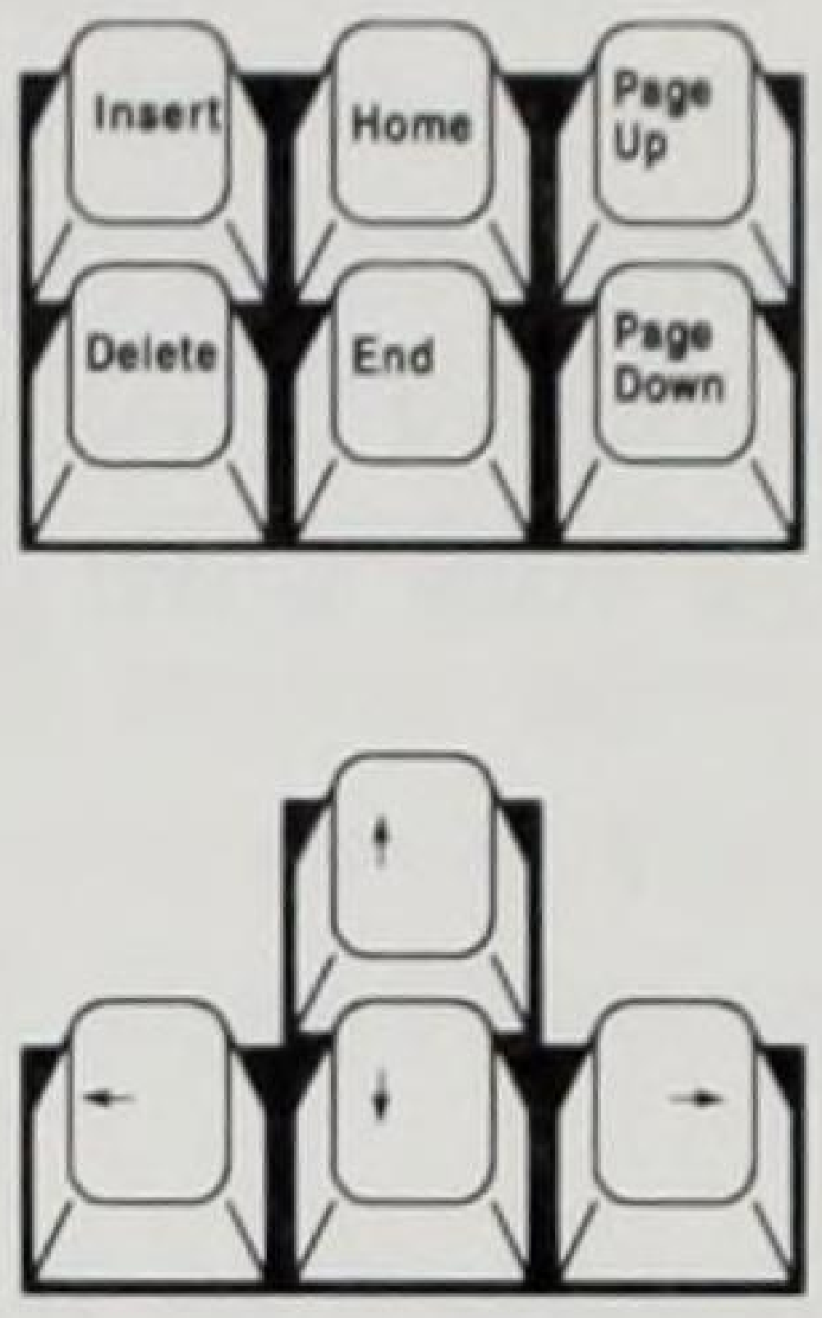


Figure 5-4. *Cursor-control keys*

If your keyboard only contains one set of arrow keys, you will need to use the NUMLOCK key as just discussed to toggle between the numeric keypad and the cursor-control keys.

Key Combinations

A *key combination* is a sequence of two or more keys you press to achieve a specific result. This section discusses the two most common key combinations. You will use the CTRL and ALT keys in combination with other keys to achieve differing results. For example, by holding down the CTRL key and pressing the C key, you can end a DOS command. Many DOS books and software manuals represent key combinations by listing the necessary keys separated by hyphens. For example, CTRL-C represents the key combination you can use to end DOS commands. In the case of CTRL-C, you would first hold down the CTRL key, and then, while holding down CTRL, press C (resulting in both keys being depressed).

Using CTRL-C to End a Command

Should you inadvertently invoke the wrong DOS command, DOS lets you cancel the command using the CTRL-C key combination. For example, suppose you invoke the DISKCOPY command (described in Chapter 12) to copy the contents of one floppy disk to another blank floppy disk, only to find you don't have an unused floppy. DISKCOPY, unfortunately, will prompt you to insert the floppy disks into drives A and B, and then to press any key to continue, as shown here:

```
C:\> DISKCOPY A: B: <ENTER>
Insert SOURCE diskette in drive A:

Insert TARGET diskette in drive B:

Press any key to continue . . .
```

Because you don't want DISKCOPY to continue, you need a way to end the command. Fortunately for you, the CTRL-C key combination directs DOS to cancel the current command. In this case, when you press CTRL-C at the DISKCOPY prompt, DOS will display ^C (control C) and will end the command, returning control to the DOS prompt:

```
C:\> DISKCOPY A: B: <ENTER>
Insert SOURCE diskette in drive A:

Insert TARGET diskette in drive B:

Press any key to continue . . .
^C

C:\>
```

Treat the CTRL-C key combination with extreme caution. If you use CTRL-C to end a command that updates one or more files on your disk, you might lose a portion of the file's contents. As a rule, you can generally end a DOS command using CTRL-C, but you should not use CTRL-C to end an application such as your word processor or spreadsheet. In fact, most applications won't let you do so. DOS also lets you use the CTRL-BREAK key combination to end commands, just as you would CTRL-C. If you try unsuccessfully to end a command using CTRL-C, try using CTRL-BREAK. Some commands might intentionally ignore CTRL-C but not CTRL-BREAK.

WARNING

Using CTRL-ALT-DEL to Restart DOS

In Chapters 18 and 24, you will learn about two special files, AUTOEXEC.BAT and CONFIG.SYS, that DOS uses during its startup process. As you work with DOS, you will need to make changes to these two files. Each time you make such a change, you need to restart DOS for the changes to take effect. Until now, your only way to restart DOS has been a *cold boot,* to turn your computer off and back on or press the Reset button. DOS, however, lets you do a *warm boot,* use the CTRL-ALT-DEL key combination to restart DOS. When you press CTRL-ALT-DEL, your screen will clear and your computer will reload DOS from disk, as if you had turned on your computer.

As a rule, only press the CTRL-ALT-DEL key combination when DOS is displaying its prompt. If you restart DOS while a program is updating a file or your disk, you might damage the file, losing the information it contains.

Depending on the number of applications you use, you may eventually encounter a program that contains an error. When you run the program, your computer appears to freeze up and doesn't do anything. In such cases you may need to use CTRL-ALT-DEL to restart DOS. If DOS does not respond to CTRL-ALT-DEL, you will need to turn your computer's power off and back on, restarting your computer's power-on sequence.

SUMMARY

Key Combinations

A key combination is a sequence of keys that you must hold down to achieve a specific result. Many books represent key combinations by separating the keystrokes with the hyphen (-). For example, the CTRL-C key combination instructs you to first hold down the CTRL key and then press C.

F3 Function Key

Each time you execute a command, DOS temporarily saves the command in case you want to repeat or edit it. Pressing the F3 key at the DOS prompt directs DOS to display the previous command for editing or execution.

F1 Function Key

The F1 function key, like F3, lets you edit or repeat the previous command. Unlike F3, which displays the entire previous command at one time, F1 displays it one letter at a time.

SUMMARY

F6 Function Key

The F6 function key lets you mark the end of the file you are copying from the keyboard. When you press the F6 key, DOS will display its end-of-file marker, ^Z (control Z).

CTRL-C Key Combination

The CTRL-C key combination lets you end a DOS command. When you press CTRL-C, DOS will display ^C (control C) on your screen, ending the application and returning control to the DOS prompt.

CTRL-ALT-DEL Key Combination

Ctrl
Alt
Del

Press the CTRL-ALT-DEL key combination to restart DOS. Use this key combination to restart DOS after you change the system startup files CONFIG.SYS and AUTOEXEC.BAT, or when your system hangs due to a program error.

Hands On

From the DOS prompt, issue the following DIR command to display the files residing in the DOS directory:

```
C:\> DIR \DOS <ENTER>
```

When the DIR command is completed, press the F3 function key followed by ENTER to repeat the command.

Next, issue the command in the following example, mistyping CLS as "CKS".

```
C:\> CKS <ENTER>
```

To correct the command, press F1, type L, and press F1 again. Press ENTER to execute the command.

You have learned that the CTRL-C key combination lets you end a command. In this case, invoke DATE:

```
C:\> DATE <ENTER>
```

When DATE displays its prompt for you to enter a new date, hold down the CTRL key and press C. DOS will end the command displaying ^C:

```
C:\> DATE <ENTER>
Current date is Mon 04-05-1993
ENTER new date (mm-dd-yy): ^C

C:\>
```

Creating short files directly from the keyboard is very convenient. Using the COPY command and the F6 function key, create the file NUMBERS.DAT, as shown here:

```
C:\> COPY CON NUMBERS.DAT <ENTER>
1 <ENTER>
2 <ENTER>
3 <ENTER>
4 <ENTER>
5 <ENTER>
^Z <ENTER>
   1 File(s) copied
```

As you will recall, you must press the F6 key or the CTRL-Z key combination to create the DOS end-of-file marker (^Z).

Using the DIR command, you can verify that the file exists on your disk as follows:

```
C:\> DIR NUMBERS.DAT <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1C56-17ED
```

```
Directory of c:\

NUMBERS DAT      15 02-27-93  10:51a
   1 File(s)  17089920 bytes free
```

Likewise, using the TYPE command discussed in Chapter 8, you can display the file's contents:

```
C:\> TYPE NUMBERS.DAT <ENTER>
1
2
3
4
5

C:\>
```

Use the following DEL command to erase the file from your disk.

```
C:\> DEL NUMBERS.DAT <ENTER>
```

Last, use the CTRL-ALT-DEL key combination to restart DOS. Your computer will follow the same sequence discussed in Chapter 2, with the exception that it won't execute its power-on self-test. If you are using a hard disk system, make sure you remove the floppy disk from drive A, if one is present, before you perform the CTRL-ALT-DEL key combination to restart DOS.

Review

1. What are the four sections common to all keyboards?
2. What is a key combination?
3. What are the functions of the following key combinations?

   CTRL-C

   CTRL-ALT-DEL
4. How does the F1 function key differ from F3?
5. What is the function of the NUMLOCK KEY?

Answers

1. All keyboards share

   Standard typewriter keys
   Function keys
   Numeric keypad
   Cursor-control keys

2. A key combination is two or more keys you hold down to achieve a specific purpose. DOS predefines several key combinations. The CTRL-C key combination, for example, directs DOS to cancel the current command. When you press the keys for a specific combination, you should hold down the keys in the order specified. For example, using CTRL-C, you should first hold down the CTRL key. Next, while you depress the CTRL key, press the C key.

3. The following key combinations have these functions:

CTRL-C	Directs DOS to cancel the current command
CTRL-ALT-DEL	Restarts DOS

4. The F1 and F3 function keys let you edit the previous command. The F1 function key retrieves the previous command a single letter at a time. The F3 function key, on the other hand, retrieves the entire command all at once.

5. The NUMLOCK key is a toggle that lets you select either the entry of numeric values or use of the cursor control keys using the numeric keypad. The first time you press NUMLOCK, your computer will use the numeric values. The second time you press NUMLOCK, your computer will activate the cursor control keys. Most keyboards have a small light that, when lit, indicates that you have selected the numeric keys.

Advanced Concepts

Much of the first section of this chapter dealt with using the F1 and F3 function keys to repeat or edit the previous command. If you are using DOS 5 or later, DOS provides the very powerful DOSKEY command that lets you recall a list of your most recently used commands. The actual number of commands DOSKEY can track depends on the amount of memory you allocate for DOSKEY as well as the various lengths of the commands. Chapter 27 looks at the DOSKEY command in detail. You might want to skip ahead to that chapter next.

The remainder of this chapter examines the function keys F2, F4, and F5, as well as the CTRL-S key combination. If you are interested in defining your own function keys, Chapter 34 discusses how to do so using the ANSI.SYS device driver.

Advanced Command-Line Editing

As you know, the F1 function key lets you copy a single letter of the previous command, while F3 copies the entire command line. Depending on the length of your command lines, you may find it faster to edit the previous command using the F2 or F4 function keys discussed here.

The F2 function key directs DOS to copy characters from the previous command line, up to, but not including, the character you type immediately after pressing F2.

For example, assume you have just issued the following DIR command:

```
C:\> DIR \DOS <ENTER>
```

Next, assuming you want to repeat the DIR command, but this time for the current directory, you can press F2 followed by the backslash character (\). DOS will copy the contents of the previous command, up to, but not including, the backslash resulting in the letters DIR:

```
C:\> DIR
```

The F4 function key directs DOS to copy the characters that include and follow the character you type immediately after pressing F4. After you type the desired letter, you must press F1 or F3 to perform the actual copy. Many users, for example, use the DIR command to search a directory for a specific command and then execute the command if it exists. The following command, for example, uses DIR to search the DOS directory for the DISKCOPY command:

```
C:\> DIR \DOS\DISKCOPY <ENTER>
```

Assuming DIR successfully locates DISKCOPY, you can quickly execute the command by pressing F4 followed by the backslash key (\) and F3. In this case, DOS will copy the characters including and following the first backslash:

```
C:\> \DOS\DISKCOPY_
```

As you have learned, the function keys F1 through F4 let you edit the previous command. If you realize you have made a mistake in the command you are currently typing, you can press F5 to edit the command using the other function keys. For example, assume you mistyped the FORMAT command as shown, but have not yet pressed ENTER.

```
C:\> FOMAT A:
```

If you press the F5 function key, DOS will display the character that indicates it has copied the current command into the previous command buffer:

```
C:\> FOMAT A:@
```

You can then press the F1 key twice displaying the letters "FO":

```
C:\> FOMAT A:@
     FO
```

Next, press the INS key to inform DOS you need to insert a letter, type **R**, and press F3 to complete the command, as shown here:

```
C:\> FOMAT A:@
     FORMAT A:
```

When you press ENTER, DOS will invoke the command as desired.

Canceling a Command with ESC

As you have learned, you can terminate a command using the CTRL-C key combination. If, however, you have typed a command's name and realize you don't want to execute the command, you can press the ESC key. DOS will respond by displaying a backslash at the end of the command and moving the cursor to the next line (which column it goes in depends on your version of DOS), as shown here:

```
C:\> DISKCOPY A: B:\
     _
```

Assuming you are using a DOS version prior to 5.0, the following example uses the ESC key to cancel the DISKCOPY command, and then CLS to clear the screen.

```
C:\> DISKCOPY A: B:\
     CLS <ENTER>
```

Using ESC in this way, you don't have to erase the entire command using the BACKSPACE key.

If you are using DOS 5 or later, pressing ESC directs DOS to erase the current command, letting you type in the next command as if you had pressed F5 to edit the current command. Many applications use ESC to cancel menus or various options also. Locate the ESC key on your keyboard. Depending on your applications, you may use it quite frequently.

Using CTRL-S to Suspend Scrolling

Depending on your computer's speed, there may be times when DOS scrolls a program's output off the top of your screen faster than you can read it. In such cases, you can direct DOS to temporarily suspend scrolling by pressing the CTRL-S key combination. To resume scrolling, simply press a key.

SUMMARY

F2 Function Key

The F2 function key directs DOS to copy characters from the previous command line, up to, but not including, the first incidence of the character you immediately type after pressing F2. If the character specified does not exist in the previous command, DOS does not copy any characters.

F4 Function Key

The F4 function key directs DOS to copy all the characters from the previous command that include and follow the characters you type immediately after pressing F4. After you type the desired letter, you must press F3 to perform the actual copy. If the character specified does not exist, F4 copies the entire command.

SUMMARY

F5 Function Key

The F5 function key directs DOS to copy the current command line to the previous command-line buffer for editing. When you press F5, DOS will display the @ character at the end of the current command line and then let you edit the command line using the other function keys.

ESC Key

The ESC key lets you cancel a command you have typed at the DOS prompt, but not yet invoked by pressing ENTER. When DOS encounters the ESC key, DOS will display the backslash (\) at the end of the command and redisplay its prompt for subsequent commands, as shown here:

```
C:\> DISKCOPY A: B:\
     _
```

You must press ESC to cancel a command in this way. You cannot simply press the backslash key.

CTRL-S Key Combination

The CTRL-S key combination directs DOS to temporarily suspend output scrolling, letting you view a command's output on the screen. To resume scrolling, simply press any key.

Key Terms

Key combination A sequence of two or more keys held down for a specific result.

Warm boot The process of restarting DOS with the CTRL-ALT-DEL key combination.

Cold boot The process of starting DOS by turning on your computer.

Chapter 6

Getting Around Your Disk

As you learned in Chapter 1, all personal computers have at least one floppy disk drive, named drive A. In addition, most computers have a hard disk named drive C and possibly a second floppy disk, drive B. The purpose of these disk drives is to let you access files stored on different disks. As you learned in Chapter 2, each time DOS starts, it selects the boot disk as its current (or default) drive. In most cases, your DOS prompt contains the drive letter corresponding to the current drive. The current drive is so named because, by default, it is the drive whose disk DOS will search for files and external commands unless you tell it to look elsewhere. In this chapter you will learn several different ways to access files stored on disks in drives other than the default.

Changing the Default Drive

The following discussion assumes you have a floppy disk that contains files in drive A, and that the drive latch is closed.

Your disk drives have single-letter names such as A, B, and C. When you specify a disk drive to DOS, you must place a colon after the disk drive name (such as A: or B:). To change from one drive to another, simply type the desired drive name (and colon) at the DOS prompt and press ENTER. In the following example, the command changes the default drive from C to A.

```
C:\> A: <ENTER>

A:\>
```

Note that DOS immediately changes the system prompt to indicate the new default drive. Depending on the commands your computer executes when it starts, your prompt may differ slightly from those shown here. Don't be alarmed—in Chapter 10, you will learn how to use the PROMPT command to modify your DOS prompt. If you issue a DIR command, DOS will now display the files that reside on the disk in drive A.

```
A:\> DIR <ENTER>
```

In other words, drive A is now the default drive. To change the default drive back to drive C, simply issue the following command.

```
A:\> C: <ENTER>
```

If your computer has a floppy disk drive B, and the drive contains a floppy disk, you can select drive B as the default using the same technique.

Accessing a Drive's Disk Without Changing Default Drives

As you have learned, the DIR command displays a directory listing of the files in a directory. In the previous section, you learned you can view a directory listing of the files on a different disk by first changing the default drive and then issuing the DIR command:

```
C:\> A: <ENTER>
A:\> DIR <ENTER>
```

The default drive is the drive that DOS uses unless you tell it otherwise. Most DOS commands let you include the desired disk drive letter in the command line. For example, the following DIR command directs DOS to perform a directory listing of the files stored on the disk in drive A, even though the current drive is C.

```
C:\> DIR A: <ENTER>
```

You *must* include the colon after the drive letter. If you don't include the colon, DIR assumes you want to list the directory entry for a file named A in the current drive. Because such a file probably does not exist on your disk, DIR will display an error message like the one below, indicating it did not find the file.

```
C:\> DIR A <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\

File not found

C:\>
```

Many of the DOS commands you will issue throughout the remainder of this book let you specify a disk drive in this way.

Accessing a File on Another Disk

In Chapter 4 you learned that the DIR command lets you list a directory entry for a specific file if you include the filename in DIR's command line, as shown here:

```
C:\> DIR FILENAME.EXT <ENTER>
```

If the file is on a disk other than the default, you can list the file's directory entry using this same technique by preceding the filename with the corresponding drive letter, as shown in the following command.

```
C:\> DIR A:FILENAME.EXT <ENTER>
```

Do not place spaces between the disk drive specifier (A:) and the filename. When DIR examines its command line, DIR will look on drive A for the file specified. All the DOS file commands you will examine in this book support this drive-letter:filename combination.

The Abort, Retry, Fail? Prompt

As you increase your use of floppy disk drives, you might inadvertently try to access a disk drive whose latch is not closed, or one that contains a floppy disk you have not yet prepared for use by DOS. (See Chapter 7 for information on the FORMAT command.) When such an error occurs, DOS will display an explanatory error message followed by the following prompt:

```
Abort, Retry, Fail?
```

DOS displays this prompt when it encounters an error it cannot resolve without user interaction. In this case, you must respond with the letter A, R, or F, which correspond to each option. The Abort option directs DOS to end (abort) the command responsible for the error and redisplay the DOS prompt. Assume for example, you issue a DIR command for drive A, only to discover drive A does not contain a floppy disk:

```
C:\> DIR A: <ENTER>

Not ready reading drive A
Abort, Retry, Fail?
```

By selecting the Abort option, you can end the DIR command, and return to the DOS prompt.

```
C:\> DIR A: <ENTER>

Not ready reading drive A
Abort, Retry, Fail?A

C:\>
```

The Retry option, on the other hand, gives you an opportunity to correct the situation causing the error, so you can direct DOS to repeat, from the start, the operation that encountered the error without having to retype the command. In the previous example, you could insert a floppy disk into drive A as needed and then type **R**, directing DOS to retry the directory operation.

The Fail option is the least used option. Fail directs DOS to skip the operation that encountered the error and continue the command's execution. In Chapter 12, you will learn how to copy several files from one disk to another. Assume that the disk you are copying from is an older floppy disk that is beginning to wear out. When DOS tries to read one of the files on the disk, the disk is so worn that DOS can't read it. As a result, DOS will display its "Abort, Retry, Fail?" prompt to determine how you want to handle the error. Using the Fail option, you can direct DOS to fail the operation causing the error, but to continue trying to copy the remaining files.

SUMMARY

Changing the Default Drive

Every disk drive on your system has a single-letter name, such as A, B, or C. To specify a disk drive to DOS, type the drive letter followed by a colon and press ENTER. The following command, for example, changes the default drive from C to A:

```
C:\> A: <ENTER>
```

SUMMARY

Accessing a Disk Other than the Default Drive

The current drive is the disk drive DOS uses by default to locate files and commands, unless you tell it to do otherwise. All the DOS disk commands let you specify the desired drive in their command line. For example, the following DIR command displays the files on the disk in drive A, even though the current drive is C.

```
C:\> DIR A: <ENTER>
```

Accessing a File from Another Disk

Throughout this book, you will examine many DOS file-manipulation commands that let you specify a filename in the command line. If the desired file is on a disk other than the default, precede the filename with a drive letter and colon, as shown in the following DIR example.

```
C:\> DIR A:FILENAME.EXT <ENTER>
```

Abort, Retry, Fail?

When DOS encounters a device-related error, such as an open disk drive latch or a printer off line, DOS will display the "Abort, Retry, Fail?" prompt to determine how you want to handle the error. The Abort option directs DOS to cancel the command causing the error and to return control to the DOS prompt. The Retry option directs DOS to repeat the operation after you have performed steps to correct the problem. Last, the Fail option directs DOS to fail the current operation, but to try to continue the command from the current point.

Hands On

For the following discussion, place your DOS system disk in drive A and close the drive's latch. Issue the following DIR command:

```
C:\> DIR <ENTER>
```

Unless told to do otherwise, DOS uses the default drive. In this case, because drive C is the default, DIR displays a directory listing of drive C's files. Use the following command to change the default drive from drive C to A.

```
C:\> A: <ENTER>

A:\>
```

Next, repeat the DIR command.

```
A:\> DIR <ENTER>
```

Because drive A is now the default, DIR displays the directory listing for the files on the disk in it.

Repeat the DIR command, this time including drive C in the command line.

```
A:\> DIR C: <ENTER>
```

In this case, DIR displays a directory listing for drive C as specified instead of using the default drive.

Issue the following command to select drive C as the default:

```
A:\> C: <ENTER>
```

Next, open drive A's latch and issue the following command:

```
C:\> DIR A: <ENTER>
```

Because DOS cannot access the disk in drive A, DOS will display its "Abort, Retry, Fail?" prompt. In this instance, type **A** to end the DIR command and return to the DOS prompt.

Repeat the DIR command. This time, when DOS displays the "Abort, Retry, Fail?" prompt, close drive A's latch, type **R**, and press ENTER. DOS will retry the DIR command and will succeed, because the drive is now ready for access.

Review

1. What is the default drive?
2. How do you change the default drive from A to B?
3. What is the function of the following command?

```
C:\> DIR B <ENTER>
```

4. Assuming drive C is the default, and the disk in drive A contains a file named READ.ME, what is the DIR command to display a directory listing for the file?
5. How does the Abort option differ from the Fail option in response to the DOS prompt to Abort, Retry, Fail?

Answers

1. The default or current drive is the disk drive whose disk, by default, DOS examines for the files and commands, unless you tell DOS to do otherwise. When you start your computer, DOS selects the drive it used to start as the default drive. Typically, the DOS prompt contains the drive letter that corresponds to the default drive (A> or C>).

2. To change the default drive, you simply type (at the DOS prompt) the letter of the desired drive, followed by a colon, and press ENTER

```
A:\> B: <ENTER>
```

3. Asking the function of the following command

```
C:\> DIR B <ENTER>
```

is a trick question. In this case, because the drive letter B is not followed by a colon (B:), DOS will search the current directory for a file named B. The correct DIR command to display the files on the disk in drive B is as follows:

```
C:\> DIR B: <ENTER>
```

4. The DIR command to display a directory listing for the file, assuming drive C is the default, and the disk in drive A contains a file named READ.ME, is

```
C:\> DIR A:READ.ME <ENTER>
```

5. The Abort option directs DOS to end the command that encountered the error and to return control to the DOS prompt. The Fail option, on the other hand, directs DOS to ignore the error, and if possible, to continue the command.

Advanced Concepts

As discussed in Chapter 1, different disk drives have different storage capacities. Depending on the age of your computer, you may want to upgrade to a newer, faster disk with increased storage capacity. If you are considering upgrading your disk drives, you need to also consider your disk controller card, as well as the disk's current power supply. Also, if you are adding a 3 1/2-inch drive to your system, you will need to upgrade to DOS 3.2 as a minimum for 720Kb drives, DOS 3.3 for 1.44Mb drives, and DOS 5 for 2.88Mb drives.

Disk Controller

If you open your computer's chassis, you will find your disk drive connects to a board, as shown in Figure 6-1. This board is your disk controller card, which provides your computer's interface to the disk. In general, the disk controller oversees every disk read or write operation.

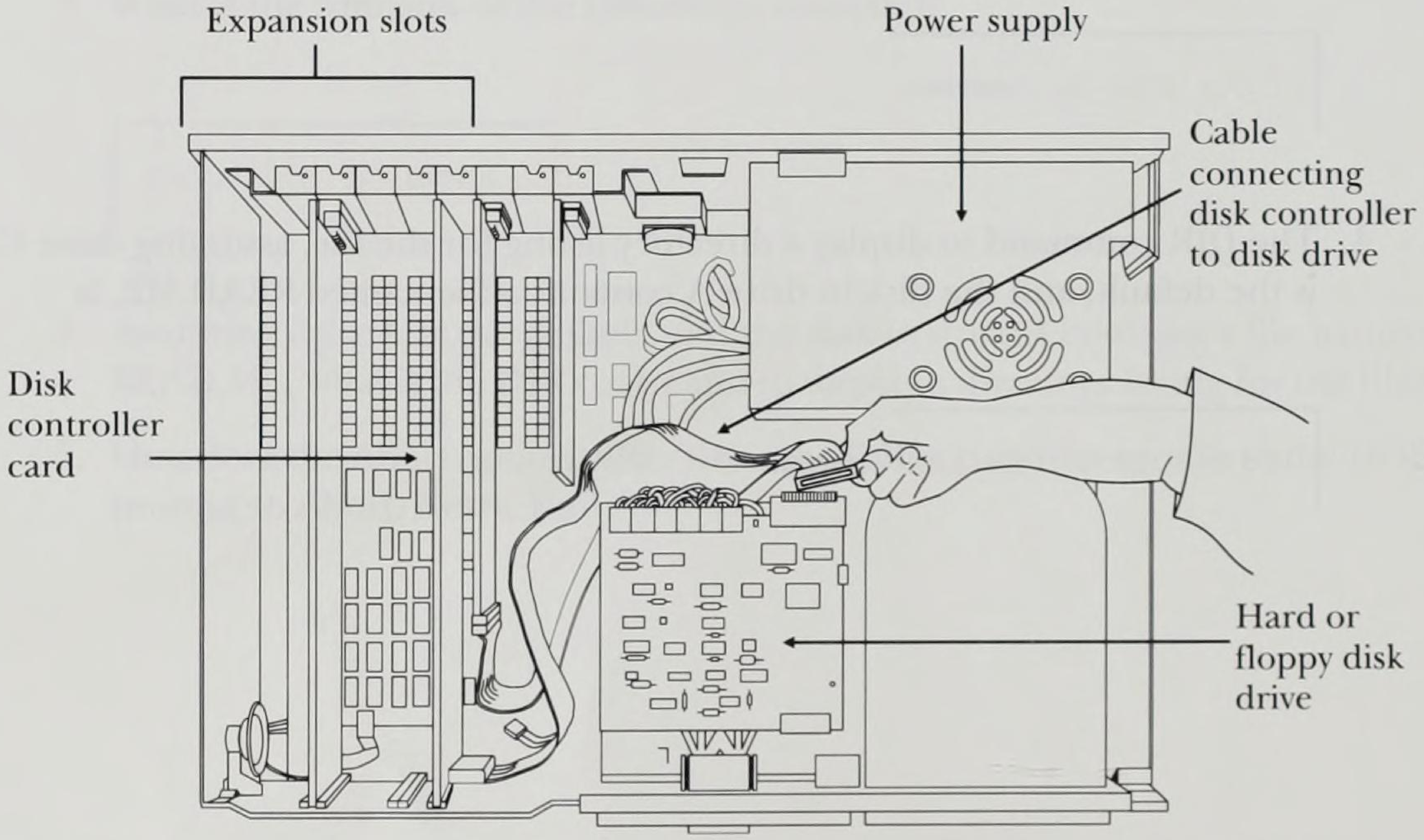

Expansion slots
Power supply
Cable connecting disk controller to disk drive
Disk controller card
Hard or floppy disk drive

Figure 6-1. *Your disk connects to a controller card*

In most cases, the disk controller is connected to both your hard and floppy disks. Before you buy a new disk, you need to ensure that the disk is compatible with your existing controller. Depending on the age of your system, you may need to upgrade the controller as well.

If you have never installed a disk yourself, I advise you to have a trained technician do the upgrade for you. Chapter 1 discusses the various disk controller types in detail.

External Floppy Disk Drives

An *external drive* is a disk drive that resides outside your computer's chassis, unlike an *internal drive*, which resides within. Depending on their computer's chassis size, some users can only add an external floppy disk drive. Many PS/2 users, for example, like to add a 5 1/4-inch floppy disk drive. Because the PS/2's chassis is so small, users must add an external disk drive. Figure 6-2 illustrates an external floppy drive.

To use an external floppy drive, you may need to install the DRIVER.SYS device driver, as discussed in Chapter 24.

If DOS displays the following error message when you attempt to access an external floppy drive, you may need to use DRIVER.SYS.

```
C:\> DIR E: <ENTER>
Invalid drive specification
```

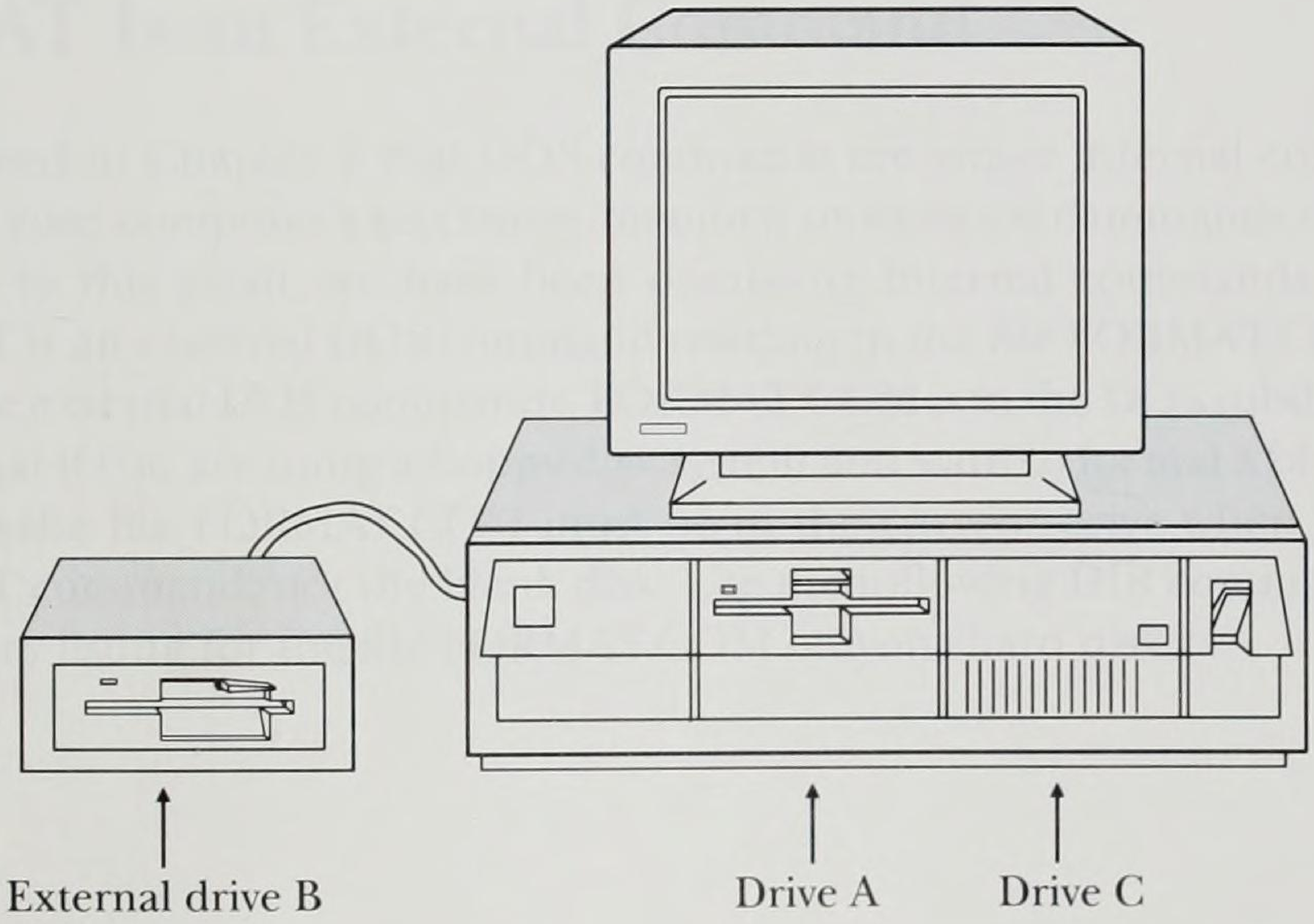


Figure 6-2. *An external floppy disk drive*

Key Terms

Default drive The default (or current drive) is the disk drive DOS will search for files or commands unless you tell DOS to look elsewhere. The DOS prompt typically contains the drive letter of the default drive.

Disk drive specifier Each disk attached to your computer has a unique single-letter name, such as A, B, or C. The disk specifier is the drive's single-letter name followed by a colon (A:).

Abort To end a command.

Disk controller The hardware device that provides the interface between a disk drive and the computer. The disk controller oversees all disk read and write operations.

External floppy disk drive A floppy disk drive connected to your computer that resides outside of the computer's chassis.

Chapter 7

Preparing Disks for Use

The examples that appear throughout this book assume you are using DOS on a computer with a hard disk. As you work with DOS and other programs, you will need to copy files from floppy disks to your hard disk or vice versa. When you buy a box of floppy disks, the disk manufacturer has no idea whether you plan to use the disks on an IBM PC or compatible using DOS, an Apple computer, or even a Commodore. Before you can copy information to a floppy disk, you must first prepare the disk for use by DOS. This process of preparing a disk for use is *formatting*. In this chapter, you will learn to use the FORMAT command to prepare a floppy disk for use.

FORMAT Is an External Command

You learned in Chapter 3 that DOS commands are either internal commands that reside in your computer's electronic memory, or external commands that reside on disk. Up to this point, we have been discussing internal commands. In contrast, FORMAT is an external DOS command residing in the file FORMAT.COM. Like the rest of the external DOS commands, FORMAT.COM is in the DOS subdirectory. This means that if you are using a floppy disk system and want to format a blank, the DOS disk with the file FORMAT.COM must be in the current drive when you issue the FORMAT command, not the blank disk. Use the following DIR command to display a directory listing for the file FORMAT.COM on your hard disk.

```
C:\> DIR \DOS\FORMAT.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45EO
 Directory of C:\DOS

FORMAT   COM     22717 02-12-93  6:00a
       1 file(s)    22717 bytes
                16762880 bytes free
```

The Formatting Process

The best way to understand the format process is to visualize an unused disk as initially blank, as in Figure 7-1.

The first stage of the format process divides your disk into circular *tracks* similar to the grooves in a record album as shown in Figure 7-2. Hard disk users refer to tracks as *cylinders*. The number of tracks on your disk depends on your disk type.

Next, FORMAT further divides each track into pie-shaped regions called *sectors*. When DOS stores a file on disk, it stores the file's contents into one or more sectors. Typically, most disk sectors store 512 bytes (or characters) of information. Figure 7-3 illustrates disk sectors.

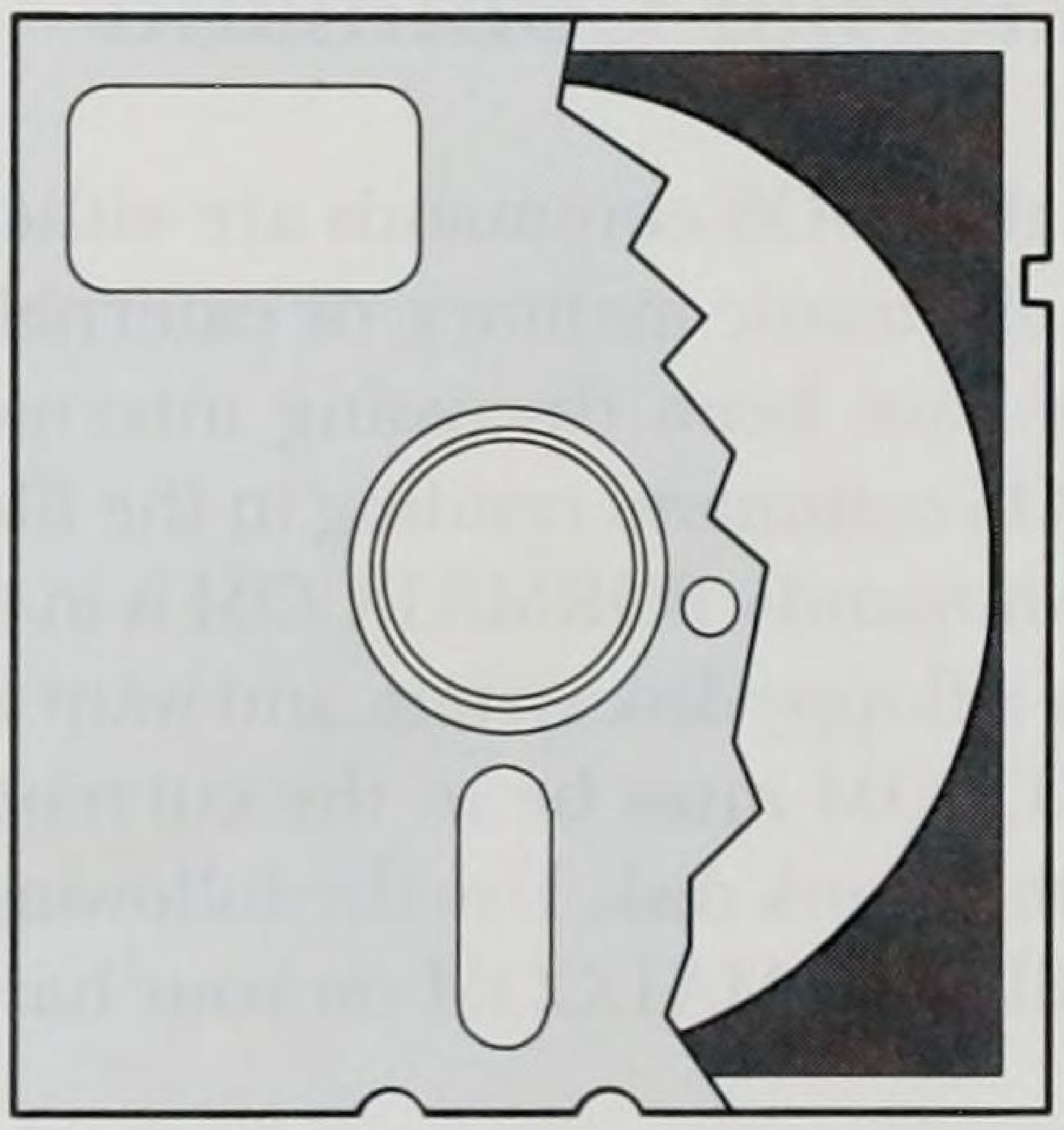


Figure 7-1. Picture an unused disk as blank

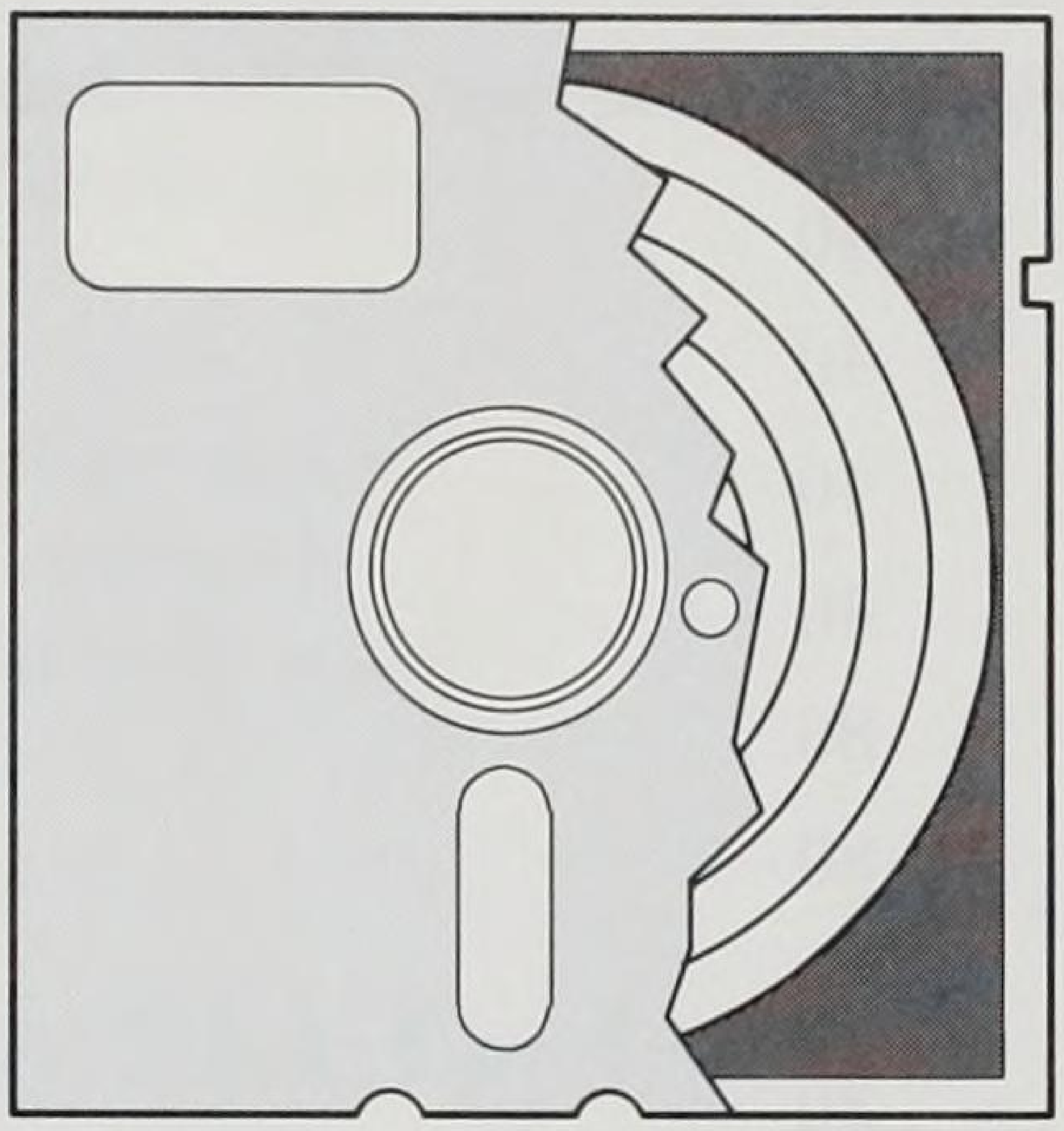


Figure 7-2. *FORMAT first divides your disk into circular tracks*

The number of sectors per track also depends on your disk type. Table 7-1 details the number of tracks and sectors per track for common disk sizes.

Knowing your disk type, you can use Table 7-1 to determine the number of tracks per side and sectors per track. Using 512-byte sectors you can determine the amount of space a disk can store using the following equation:

Storage = (Number of sides) × (Tracks per side) × (Sectors per track) × (Sector size)

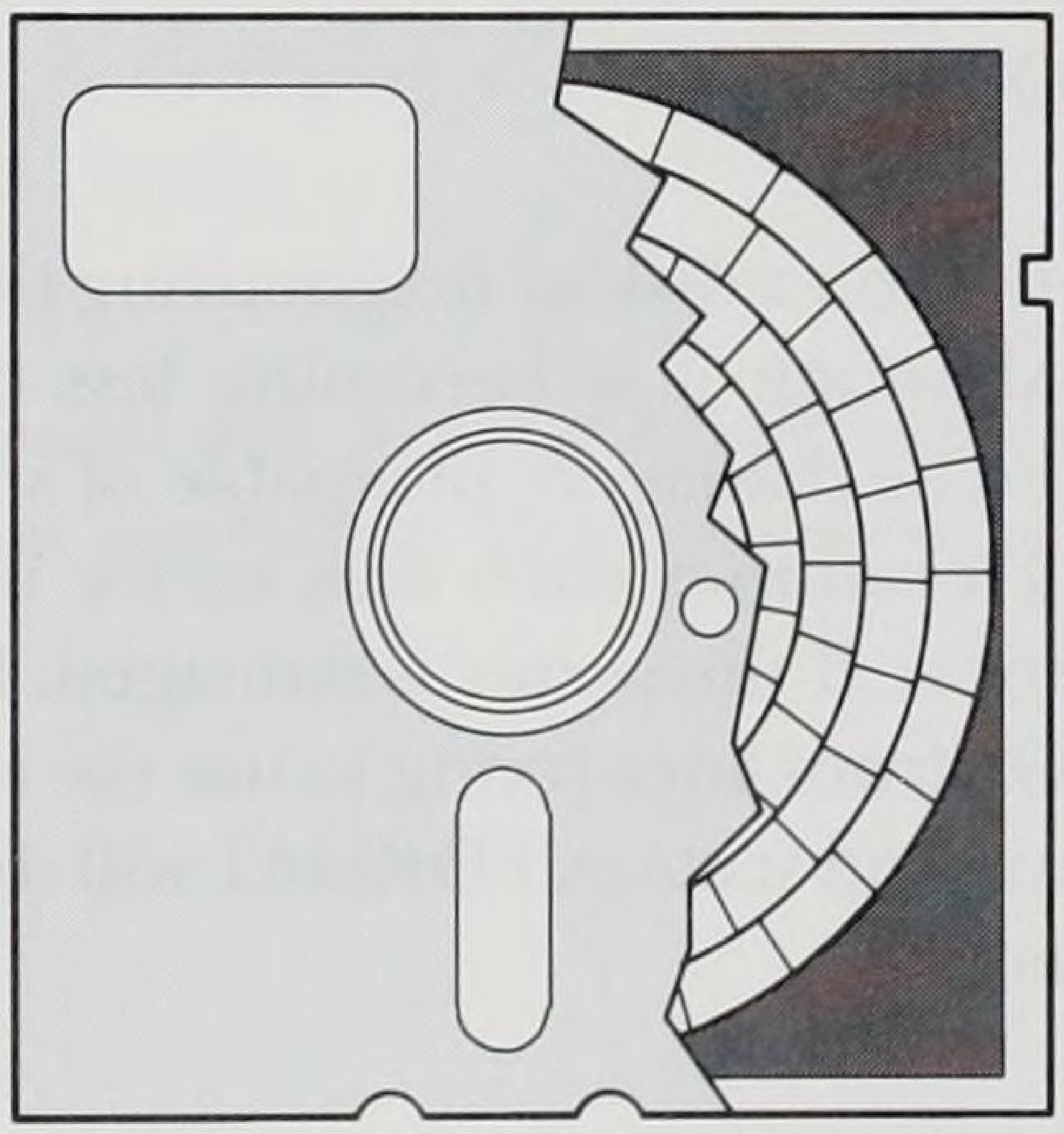


Figure 7-3. *FORMAT further divides each track into sectors*

Disk Size	Tracks per Side	Sectors per Track
160Kb	40	8
180Kb	40	9
320Kb	40	8
360Kb	40	9
720Kb	80	9
1.2Mb	80	15
1.44Mb	80	18
2.88Mb	80	36
Hard disk (10Mb)	306	17
Hard disk (20Mb)	615	17

Table 7-1. *Tracks per Side and Sectors per Track for Common Disk Sizes*

Using a 360Kb disk, for example, the disk's storage capabilities become the following:

$$\text{Storage} = (2) \times (40) \times (9) \times (512)$$
$$= 368{,}640 \text{ bytes}$$

Not All Disk Sectors Are Capable of Storing Information

A disk drive stores information on a disk by magnetizing the information to the disk's specially coated surface. Although it is becoming less common, it is possible for sections of a disk's surface to be damaged, incapable of storing information. As part of its processing, FORMAT examines each disk sector to ensure that the sector is capable of storing information. If the sector is damaged, FORMAT marks the sector as unusable, preventing DOS from later trying to use the sector to store information. If FORMAT encounters damaged sectors, FORMAT will display a message describing the number of damaged sectors

```
nnn bytes in bad sectors
```

where *nnn* represents the number of bytes in bad sectors.

The following summary informs you the disk has 5120 bytes of damaged sectors. Note that FORMAT has subtracted the number of damaged bytes from the total disk space to yield the disk's available space.

```
362496 bytes total disk space
  5120 bytes in bad sectors
357376 bytes available on disk
```

Preparing a Floppy Disk for Use

The FORMAT command prepares a disk for use by recording on the disk information DOS will later use when you store files. As a result of its processing, FORMAT overwrites any existing information the disk may contain. Should you accidentally format the wrong disk, you normally cannot retrieve any of the information the disk previously contained. When you use FORMAT, take considerable care to prevent such accidental loss of information.

To begin, place an unused floppy disk in drive A. Next, issue the following FORMAT command to prepare the disk for use.

```
C:\> FORMAT A: <ENTER>
```

FORMAT will prompt you to insert a new disk in drive A:

```
Insert new diskette for drive A:
and press Enter when ready...
```

After you insert the unused disk in drive A and close the drive latch, press ENTER to continue the disk format operation. Depending on your DOS version, FORMAT might display the message:

```
Head: n Cylinder: nn
```

or the message:

```
nn percent completed.
```

As the format process continues, FORMAT will update the messages, informing you about how much of the disk it has formatted. When FORMAT completes, it displays the following message.

```
Format complete.
```

Again, depending on your DOS version, FORMAT may prompt you to type in a name for the disk, as shown here:

```
Volume label (11 characters, ENTER for none)?
```

A disk *volume label* is simply a disk name containing up to 11 characters. The label should help you better understand the files stored on the disk. Hard disk users might assign the disk label DOS to their hard disk. For a floppy disk, you might use a more specific label, such as REPORTS, WORKSHEETS, or MEMOS93, depending on the files you are storing on the floppy disk. If you don't want to assign a volume label, simply press ENTER

Last, FORMAT will display a summary of the amount of disk space available on disk, as well as a prompt asking if you want to format additional disks.

```
362496 bytes total disk space
362496 bytes available on disk

  1024 bytes in each allocation unit.
   354 allocation units available on disk.

Volume Serial Number is 3703-10F3

Format another (Y/N)?
```

If you want to format additional disks, type **Y** and press ENTER. The format process will repeat, beginning at the prompt for you to insert a new disk into the drive. If you instead type **N** and press ENTER, FORMAT will end, and DOS will redisplay its prompt. In either case, the floppy disk in drive A is now ready for use by DOS.

Before you can understand the statistical information FORMAT displays, you need a better understanding of the actual processing FORMAT performs.

Formatting a Disk with Less Storage Capacity than the Floppy Drive

In Chapter 1 you learned that although two disks are the same size (both 5 1/4-inch or both 3 1/2-inch), the disks might not be able to store the same amount of information. When the original IBM PC was released in 1981, for example, a 5 1/4-inch floppy disk could store 163,840 bytes (160Kb). Today, 5 1/4-inch disks store up to 1,213,952 bytes (1.2Mb). A problem arises when users try to format an older, smaller-capacity disk in a newer higher-capacity disk drive. For example, if your computer has a 1.2Mb disk drive, FORMAT, by default, will try to format every disk you place into the drive as a 1.2Mb disk. Unfortunately, if you attempt to format a 360Kb disk using your 1.2Mb drive, the format will either fail or produce a disk with hundreds of unusable sectors. This same problem occurs when users try to format a 720Kb 3 1/2-inch disk as a 1.44Mb disk.

If you need to format a smaller-capacity disk in a larger-capacity drive, you must include switches in FORMAT's command line. In the case of the 360Kb disk in a 1.2Mb drive, you can specify the /4 switch, as follows:

```
C:\> FORMAT A: /4 <ENTER>
```

The actual format process is the same as that discussed earlier, with the only difference being FORMAT will use the correct disk size. If you need to format a 720Kb disk in a 1.44Mb drive, you can use the /F switch, like this:

```
C:\> FORMAT A: /F:720 <ENTER>
```

For more information on FORMAT's disk-size switches, refer to the "Command Reference" section at the end of this book.

Formatting Your Hard Disk

The majority of the discussion in this chapter has concerned formatting floppy disks, as opposed to hard disks. The reason for this is quite simple: There are only a few times when you would ever need to format your hard disk. The first occurs when you are upgrading to a higher version of DOS. The second occurs when you are experiencing hard disk errors, and the third occurs when you are correcting fragmented files, as discussed in Chapter 42.

As you have read, FORMAT *completely* overwrites the contents of the disk it formats. Regardless of why you are formatting your hard disk, always make sure you have completely backed up all the files the disk contains before performing the format.

Because an inadvertent format of your hard disk could be disastrous in terms of the information you could lose, FORMAT will display the following message, warning that you are about to format your hard disk and prompting you to ensure that you really want to continue.

```
WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE C: WILL BE LOST!
Proceed with Format (Y/N)?
```

NOTE *Should you inadvertently format your hard disk, several software utility programs exist that may be able to recover portions of your disk. If you are using DOS 5 or later, you may be able to recover your disk using the UNFORMAT command discussed in Chapter 28.*

SUMMARY

FORMAT Command

The FORMAT command lets you prepare a disk for use by DOS. To begin, FORMAT divides your disk into circular tracks, conceptually similar to the grooves in a record album. Next, FORMAT further divides each track into sectors. When DOS later stores a file on your disk, DOS will use one or more disk sectors.

WARNING *FORMAT is a destructive command, meaning FORMAT will overwrite any existing information the target disk may contain. Should you inadvertently format a disk, you may be able to recover a portion of the disk's previous information using a third-party software utility or the DOS 5 or later UNFORMAT command. Get into the habit of using DIR to examine the contents of every disk you intend to format. In so doing, you will greatly reduce the possibility of formatting a disk that contains needed files.*

Hands On

Locate an unused floppy disk and place the disk in drive A. Issue the following DIR command to ensure that the disk does not contain any files you might need.

```
C:\> DIR A: <ENTER>
```

If you have never formatted the disk, DOS will display a message similar to the following:

```
General failure reading drive A
Abort, Retry, Fail?
```

If this error message occurs, type **A** to abort the DIR command.

Next, issue this FORMAT command to prepare the disk in drive A for use.

```
C:\> FORMAT A: <ENTER>
```

If you are formatting a disk whose storage capacity is smaller than that of the disk drive, use either the /4 or /F switch, as discussed in this chapter.

If FORMAT prompts you to enter a volume label, type **TESTDISK** as follows:

```
Volume label (11 characters, ENTER for none)? TESTDISK <ENTER>
```

Last, note FORMAT's summary information to determine if the disk contains any damaged sectors. Disk quality has rapidly improved over the past few years to the point where bad disk sectors are rare. If several of your floppy disks format with damaged sectors, you should strongly consider purchasing a different brand of disk.

When the FORMAT command completes, use the following DIR command to display the contents of drive A. Note that DIR displays the disk volume name TESTDISK in its directory listing.

```
C:\> DIR A: <ENTER>
```

Review

1. Is FORMAT an internal or external DOS command?
2. List three operations FORMAT performs.
3. When do you need to use FORMAT's /4 and /F switches?
4. What is a volume label?
5. What is a bad sector?

Answers

1. FORMAT is an external DOS command that resides in the file FORMAT.COM.
2. The three operations FORMAT performs are as follows:
    - Divides your disk into tracks and sectors
    - Identifies bad sectors and marks the sectors unusable for DOS
    - Optionally assigns a disk volume label

    The "Advanced Concepts" section of this chapter discusses several other functions FORMAT performs.
3. The FORMAT /4 option lets you format a 360Kb disk in a 1.2Mb disk drive. Likewise, using the /F option, you can format a 720Kb disk in a 1.44Mb drive. The only time you need to use the /4 or /F switches is when you are formatting a disk with less storage capacity than your disk drive.
4. A volume label is an optional 11-character name you can assign to your disk. Hard disk users can assign a name like DOS to their disk. For floppy disks, you might want to use names such as REPORTS, 123-DATA, or LETTERS, depending on the type of files the disk contains.
5. A bad or damaged sector is a sector on your disk that is not capable of storing information. Information is stored on a disk by magnetizing the information onto a specially coated surface. If the surface is scratched, or slightly damaged during manufacturing, the disk cannot store information in that sector. FORMAT will mark the sector as unusable.

Advanced Concepts

FORMAT, at a basic level, divides your disk into tracks and sectors. In addition, FORMAT defines, for the target disk, several key values that directly influence the amount of disk space files consume, as well as the number of entries you can store in a disk's root directory. Further, FORMAT lets you create bootable system disks.

Building a Bootable System Disk

For DOS to start, a disk must contain two hidden system files and the file COMMAND.COM. In the case of MS-DOS, the hidden files are named IO.SYS and MSDOS.SYS. DOS hides these files to prevent their inadvertent deletion.

For the DOS system startup process to locate these two files, DOS requires that the files occupy the first two root directory entries. The best time, then, to place these files on the disk is before the root directory contains any file or directory entries, immediately after the format operation completes.

By default, FORMAT does not place the hidden system files on each disk you format. Depending on your DOS version, these files can consume up to 70Kb of disk space. Since most users don't need to put DOS with every disk, placing the hidden files on each disk would consume disk space unnecessarily.

To build a bootable system disk using FORMAT, you must include the /S qualifier, shown here:

```
C:\> FORMAT A: /S <ENTER>
```

The format process will proceed as before. However, before completing, FORMAT will transfer the system files to the disk. FORMAT will then display, in its summary messages, a message indicating it has transferred the system files, as well as the amount of disk space the system files consumed, as shown here:

```
System transferred

Volume label (11 characters, ENTER for none)? TESTDISK <ENTER>

   362496 bytes total disk space
   118784 bytes used by system
   243712 bytes available on disk
```

Depending on your DOS version, FORMAT might not copy the DOS command processor (COMMAND.COM) to the target disk. As discussed in Chapter 2, DOS needs COMMAND.COM to boot. If FORMAT did not copy the file to the disk, do so using the COPY command, like this:

```
C:\> COPY \DOS\COMMAND.COM A: <ENTER>
```

Every Disk Has a Boot Record

Whether you format a disk as a bootable system disk or not, DOS assigns the first sector on the disk for a *boot record.* The boot record contains specifics about the disk size, number of sectors, and so on. If the disk is bootable, the boot record contains a small software program that begins the DOS startup process. If the disk is not bootable, the boot record contains software that displays the following message when you try to boot from the disk:

```
Non-System disk or disk error
Replace and press any key when ready
```

Figure 7-4 illustrates the location of a boot record in the disk's first sector.

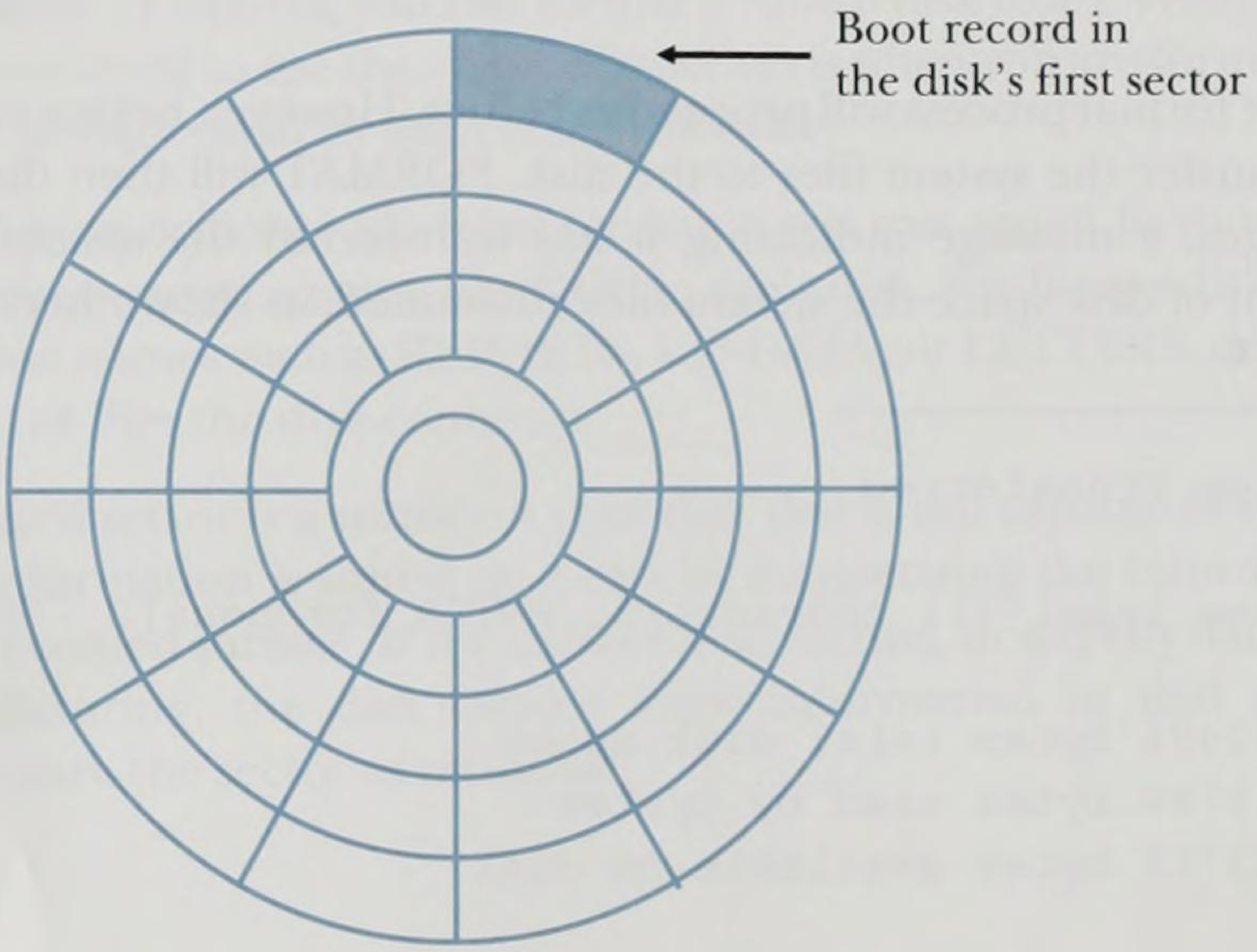


Figure 7-4. *The first sector of every disk contains a boot record*

Reserving Space for the File Allocation Table

Chapter 33 examines in detail how DOS actually stores files on your disk. In that chapter, you'll learn that DOS tracks the actual sectors used to store each file's sectors in a table called the *file allocation table* or *FAT.* DOS stores the file allocation table on your disk. Each time you perform a file operation, copying a file or executing an external command, for example, DOS uses the file allocation table to locate the file's sectors on disk.

In general, the file allocation table serves as your disk's road map. Because DOS stores the file allocation table on disk, you can imagine the impact of a disk error in one of the sectors containing the FAT. If DOS were unable to access the FAT, it would also be unable to locate any of the files on the disk. To prevent a single disk error from having such devastating effects, DOS typically stores two copies of the FAT on your disk. If DOS is unable to access the first copy of the FAT, it uses the second copy.

After FORMAT places the boot record on your disk, FORMAT places the file allocation tables on your disk. DOS stores the file allocation tables on your disk right after the boot record, as shown in Figure 7-5

Reserving Space for the Root Directory

Every DOS disk automatically starts with a root directory. Whether you format a disk as a system disk or not, one of the major functions FORMAT performs is the reservation of disk sectors for the root directory.

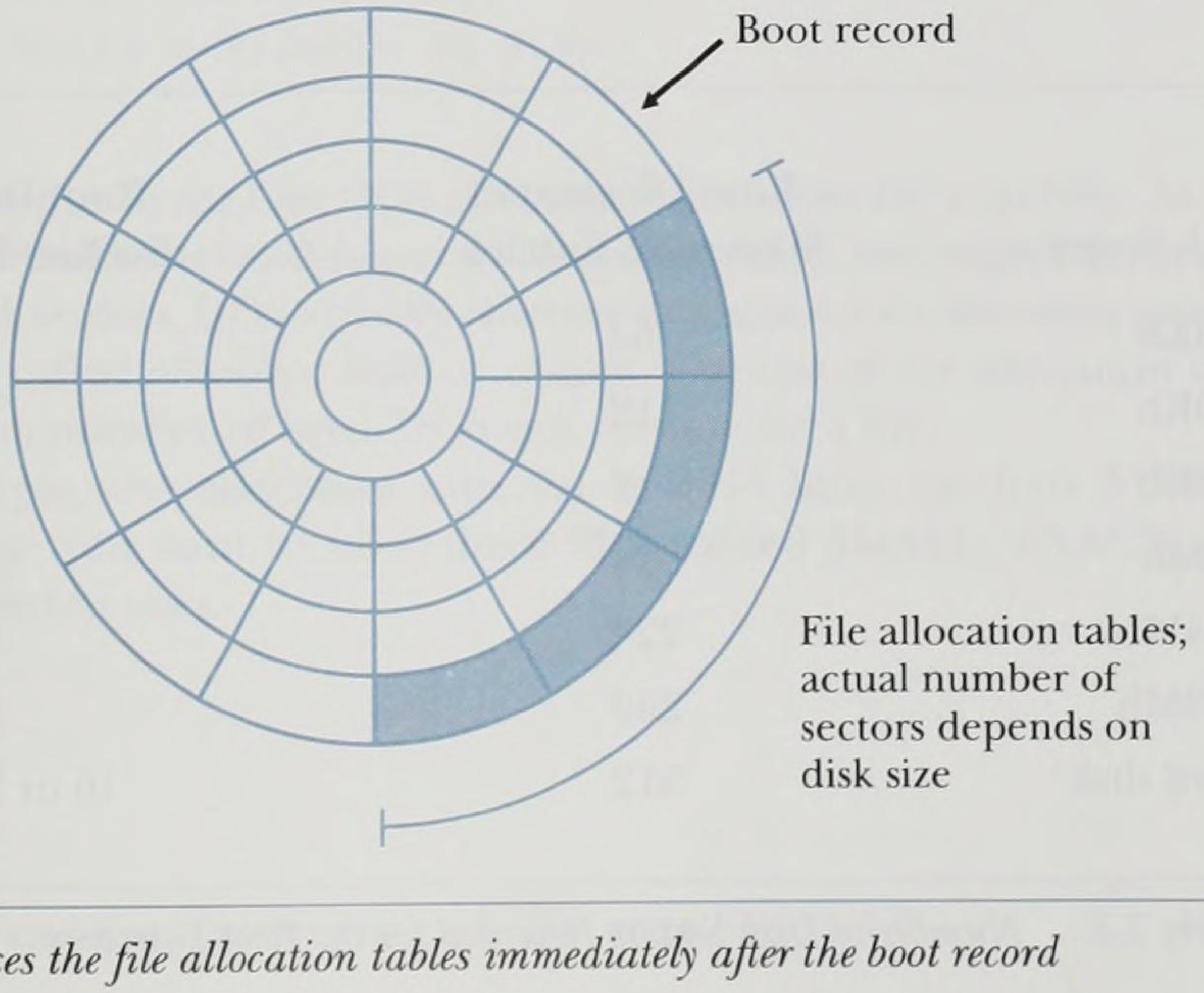

Boot record
File allocation tables; actual number of sectors depends on disk size

Figure 7-5. DOS places the file allocation tables immediately after the boot record

Each time you format a disk, FORMAT assigns sectors to serve as the root directory. The actual number of sectors DOS reserves for the root directory depends on your disk type. Table 7-2 defines the number of sectors DOS reserves for the root directory for each disk type.

FORMAT places the disk's root directory sectors immediately after the file allocation tables, as shown in Figure 7-6.

Each time you create a subdirectory or file, DOS allocates a 32-byte directory entry that contains the file or directory name, size, date and time stamps, and so on. Knowing the number of disk sectors FORMAT reserves for your disk and the disk sector size, you can determine the number of entries your root directory can store using the following equation.

Number of entries = Number of sectors × sector size ÷ 32

Once FORMAT establishes the disk's root directory, DOS cannot increase the root's size. If you try to place more files or directories in the root than DOS can store, DOS will display the following error message.

```
File creation error
```

For most users, the number of entries the root directory supports is more than enough.

Disk Space	Root Directory Maximum Entries	Root Directory Sectors Reserved
160Kb	64	4
360Kb	112	7
720Kb	112	7
1.2Mb	224	14
1.44Mb	224	14
2.88Mb	240	15
Hard disk	512	16 or 32

Table 7-2. Number of Disk Sectors Reserved for the Root Directory

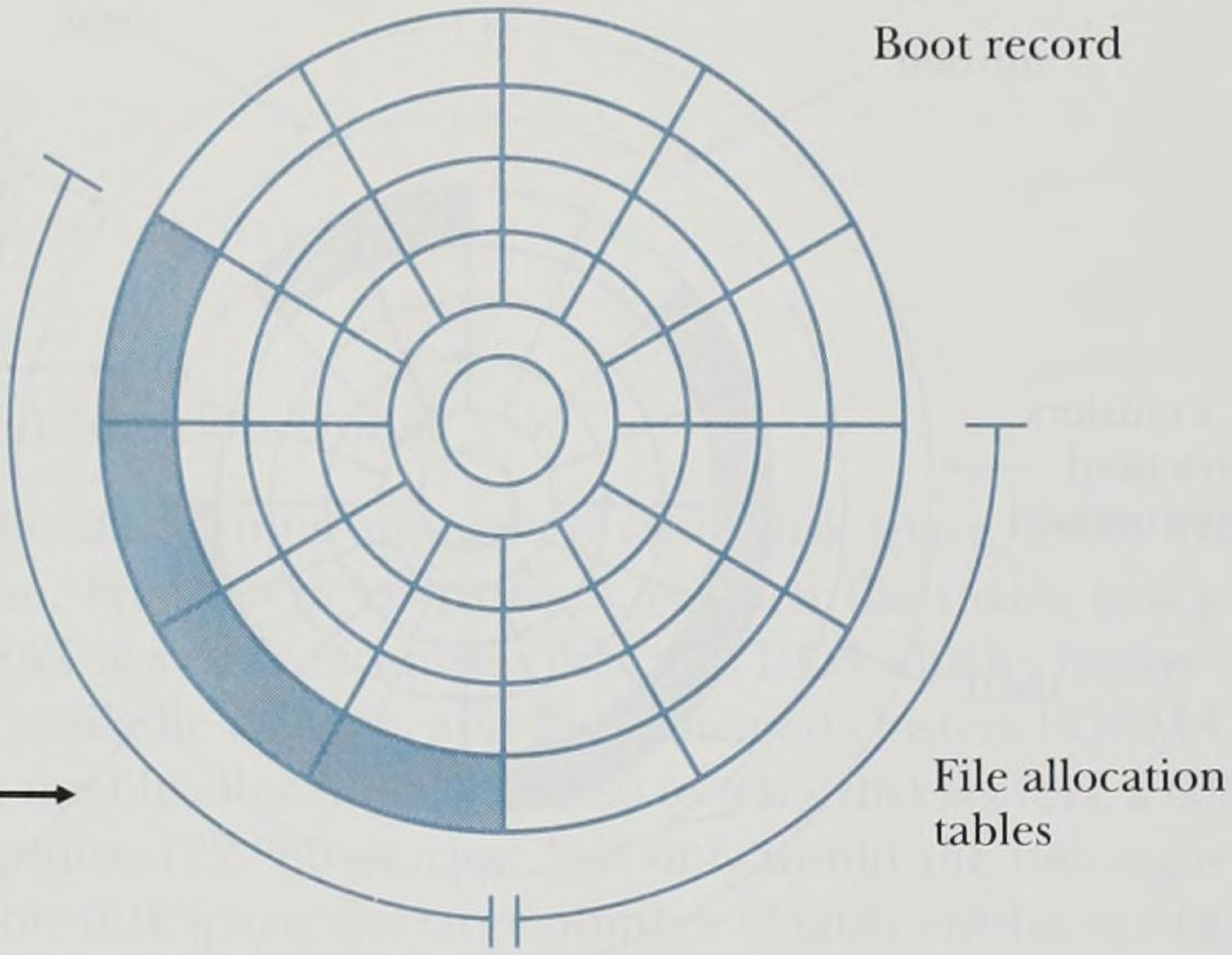

Boot record
Root directory sectors; actual number of sectors depends on disk size
File allocation tables

Figure 7-6. *FORMAT places the root directory sectors immediately after the boot record and file allocation tables*

Understanding Allocation Units

If you examine the disk statistics FORMAT displays after it successfully formats a disk, you will find a size for allocation units, as shown here:

```
1024 bytes in each allocation unit.
 238 allocation units available on disk.
```

For simplicity, we typically say that DOS allocates disk sectors for your files. As the size of a file grows, DOS allocates additional sectors as required. Although DOS stores a file's contents in disk sectors, DOS actually reserves disk space for a file using groups of connected sectors called *allocation units* or *clusters*. The size of the allocation unit tells you the minimum number of bytes DOS can allocate for a file.

Assume for example, the allocation unit size is 2048 bytes (or four 512-byte sectors). Next, assume you want to store three files named SMALL, EXACT, and LARGE with the following sizes.

```
SMALL  1024 bytes
EXACT  2048 bytes
LARGE  3072 bytes
```

DOS will allocate and use clusters for each file, as in Figure 7-7

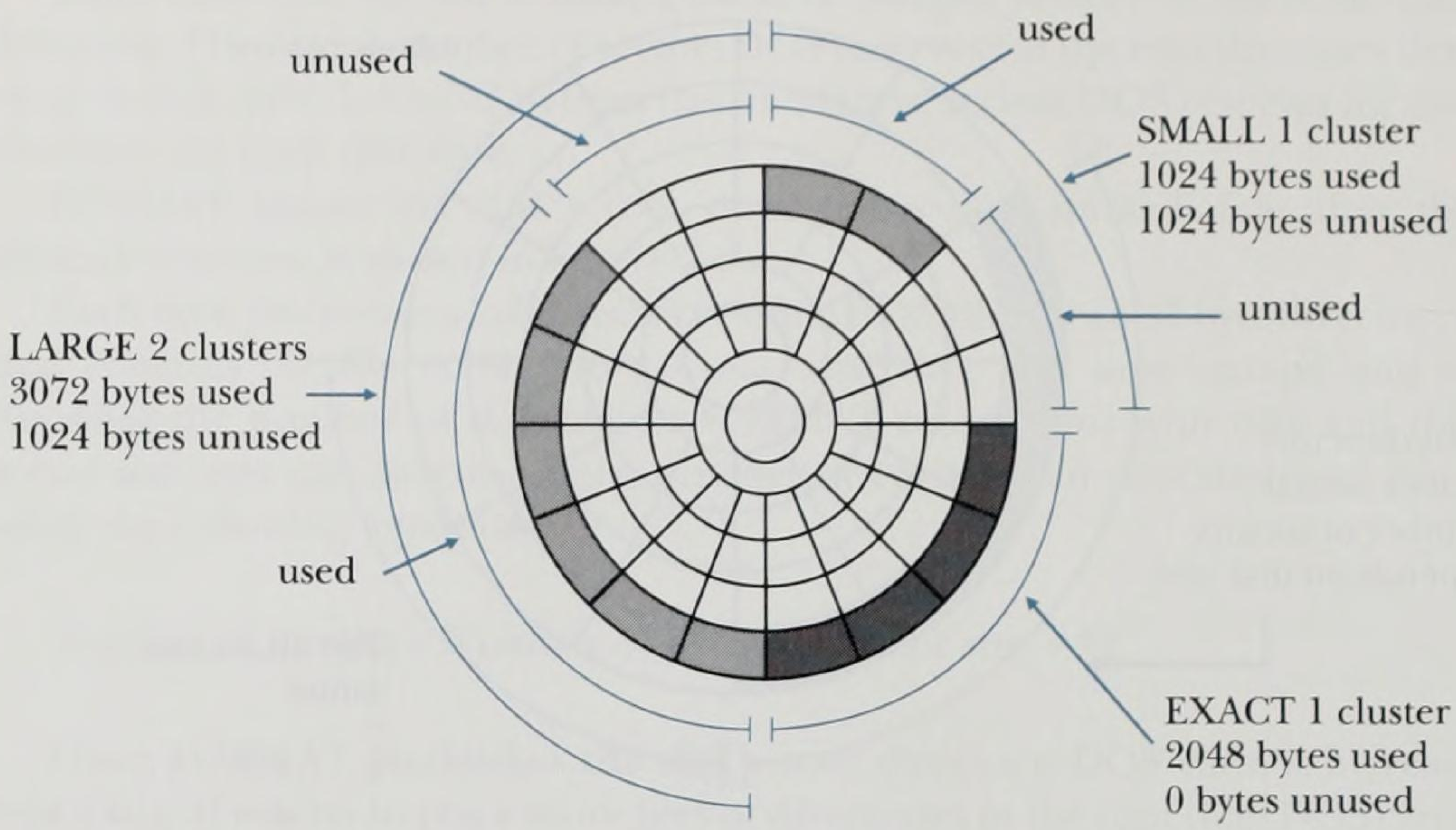

unused
used
SMALL 1 cluster
1024 bytes used
1024 bytes unused
unused
LARGE 2 clusters
3072 bytes used
1024 bytes unused
used
EXACT 1 cluster
2048 bytes used
0 bytes unused

Figure 7-7. *DOS allocates disk space in clusters*

Remember the allocation unit dictates the smallest amount of disk space DOS can allocate for a file. Although the file SMALL only has 1024 bytes of information, DOS must still allocate a complete allocation unit for the file, or in this case, 2048 bytes leaving 1024 bytes unused. If the file size grows in the future, DOS will use the available space. If the file size never changes, the disk space will remain unused and inaccessible to other files.

As an example, perform a directory listing and note the amount of available disk space displayed.

```
15 File(s) 102400 bytes free
```

Next, create the file ONE as follows:

```
C:\> COPY CON ONE
1^Z <ENTER>
     1 File(s) copied
```

In this case, the file only contains one byte of information. However, DOS must still reserve a complete allocation unit for the file resulting in 2047 bytes of unused disk space. If you repeat the directory command, you will find DOS has decreased the amount of available disk space by an allocation unit, which in this example is 2048 bytes.

```
16 File(s) 100352 bytes free
```

Why Does DOS Use Allocation Units?

Although using file allocation units appears to waste disk space, DOS has several reasons for using them. First, DOS keeps track of each file's data using the file allocation table. The FAT is a table of entries that tells DOS which clusters each file is using, as well as the available clusters, and the damaged clusters FORMAT marks as unusable. Assuming the file allocation table had to track disk sectors, a 60Mb disk, for example, would require 122,880 entries! Not only would the two copies of the FAT take up considerable disk space, the large number of table entries would become very time consuming for DOS to traverse, which would decrease your system performance. Second, in Chapter 42, you will learn how disk fragmentation can decrease your system performance. Disk fragmentation occurs when DOS stores a file's contents in locations across your disk, as opposed to consecutive storage locations.

Figure 7-8 illustrates two files. The first file's contents are stored in consecutive locations, while the second file's contents are spread out across the disk.

When DOS accesses the contiguous file, DOS can read the file's contents one sector after another. For the fragmented file, however, DOS must wait for the disk to

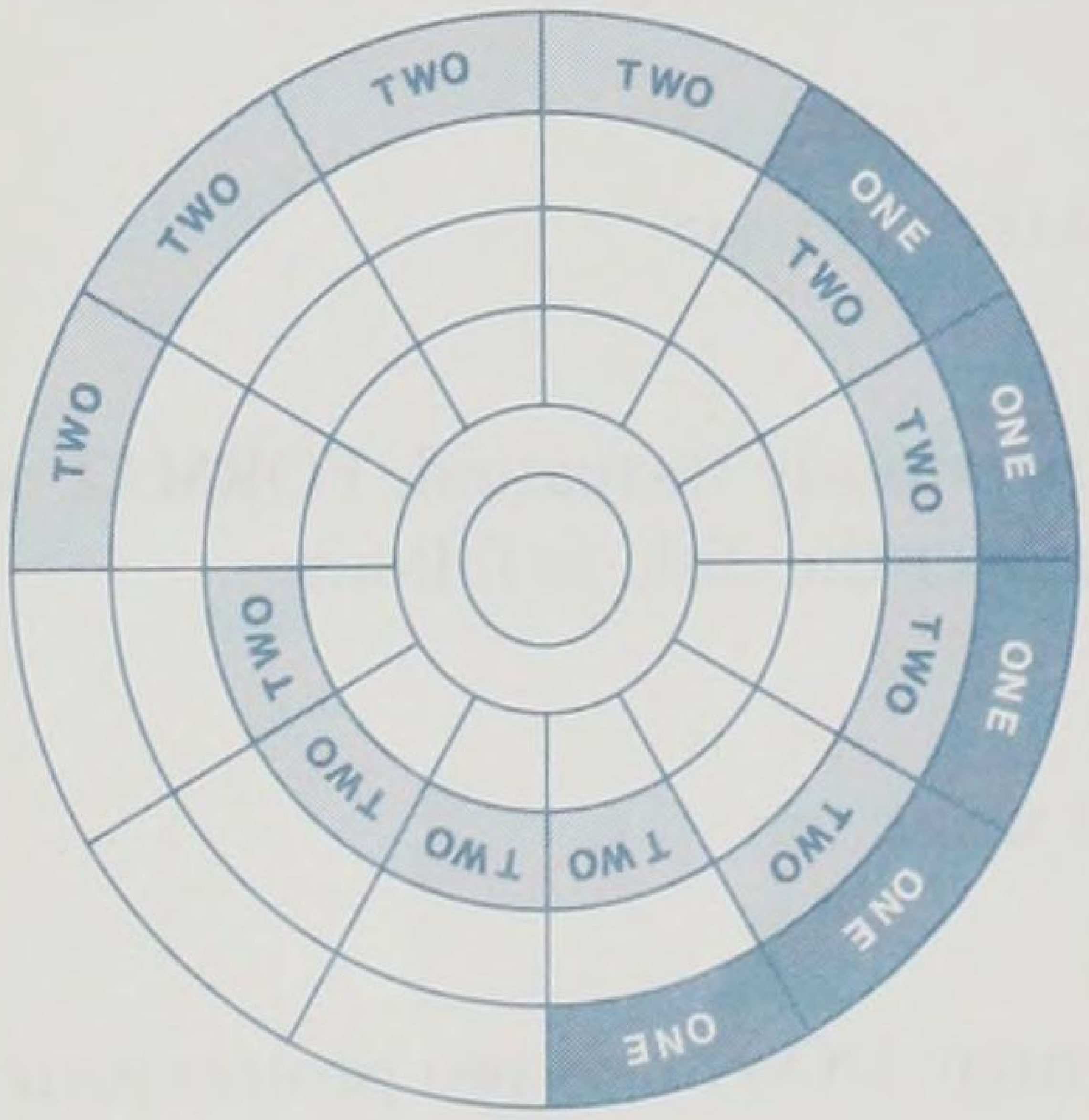


Figure 7-8. A contiguous file (ONE) and a fragmented file (TWO)

rotate each sector of the file under the disk read/write head. These rotational delays decrease your system's performance.

By reserving disk space for your files using clusters, DOS decreases the chance of disk fragmentation by grouping multiple sectors of the file together. Disk fragmentation is most common when a file's size increases, because DOS might have to allocate additional disk space for the file. By using clusters, DOS can first allocate the unused portion of the cluster before having to look elsewhere on the disk for storage locations.

Chapter 33 examines how DOS stores files on disk and how the file allocation table tracks the file locations. For now, simply understand that, no matter how hard you try, the allocation unit size determines the smallest amount of disk space DOS will reserve for a file. If you think you are saving disk space by creating several very small files, in reality you're not.

Formatting in DOS 5 and 6

Chapter 28 discusses the UNFORMAT command in detail. In general, should you accidentally format the wrong disk, the UNFORMAT command may be able to retrieve the disk's information. Unless you tell FORMAT to do otherwise, by including the /U switch, FORMAT will perform a *safe format* of your disk. A safe format creates a new root directory and initializes the file allocation table. A safe format does not, however, overwrite the information that currently resides on the disk. In addition, before the safe format initializes the file allocation table and root directory, the safe format makes a copy of both, which it stores on the disk. Should you inadvertently format the wrong disk, UNFORMAT can use the information the safe format stores on the disk to retrieve the disk's previous information.

When you format a disk using DOS 5 or later, FORMAT will check if the disk has been previously formatted and will display the following:

```
Checking existing disk format
```

If the disk has been previously formatted, FORMAT will save the information UNFORMAT needs to recover the disk, as follows:

```
Saving UNFORMAT information
```

By providing a safe format, DOS helps you protect your disks and files.

Key Terms

Disk formatting The process of preparing a disk for use by a specific operating system.

Tracks Circular divisions of your disk that DOS uses to locate specific disk sectors. Hard disk users often use the term cylinders instead of tracks.

Sectors Storage regions on your disk. DOS stores your files in one or more sectors.

Allocation unit The smallest amount of disk space DOS can allocate for a file. Allocation units are also called clusters.

Fragmented file A file whose contents are stored in locations spread out across a disk. Fragmented files decrease system performance.

Volume label An 11-character name you can assign to your disk. Volume labels aid in floppy disk organization.

Bad sector A damaged disk sector, incapable of storing information.

File allocation table (FAT) A table DOS uses to track the location of each file's contents, as well as the available disk space and damaged sectors. The file allocation table is your disk's road map.

Chapter 8

Fundamental File Operations

DOS organizes information on your disk in files. Consider your disk files as if they were files stored in paper folders in an office. With paper files, you examine the files' contents, make copies of files, rename files as necessary, and discard the files when they are no longer needed. As you will learn in this chapter, DOS provides four fundamental file commands:

TYPE	View a file's contents
COPY	Make a duplicate of a file
RENAME	Change a file's name
DEL	Delete a file that is no longer needed

These four commands provide you with the key file-manipulation capabilities you will use every day. Because of their frequent use, all four of these commands are internal commands, and DOS keeps them in your computer's electronic memory at all times. If you are using DOS 6, the MOVE command discussed in Chapter 11 lets you move a file from one directory to another.

TYPE Displays a File's Contents

Just as you browse through the contents of paper files, so will you occasionally want to browse through files stored on disk. The TYPE command lets you display a file's contents, provided the file contains standard text (characters and numbers). For example, the following TYPE command displays the contents of a file named AUTOEXEC.BAT:

```
C:\> TYPE  \AUTOEXEC.BAT <ENTER>
PATH C:\DOS;C:\DISKUTIL
PROMPT $P$G
```

Depending on the configuration of your system, the file contents of your AUTOEXEC.BAT may differ from those shown. If TYPE displays the "File not found" message, your disk might not yet have an AUTOEXEC.BAT file. Don't worry; you will create an AUTOEXEC.BAT file in Chapter 18.

You learned in Chapter 4 that DOS files can store external commands, spreadsheets, word processing documents, and so on. TYPE cannot meaningfully display some of these on your screen. If you try to view the contents of a command file having the COM or EXE extension, for example, your computer will very likely beep and display unrecognizable characters on your screen. Likewise, if you use TYPE to view a file containing a word processing document, the file's contents may again be difficult to read. Word processors let you produce professional-quality documents by letting you align the left and right margins, type in boldface text, and so on. To accomplish these capabilities, word processors embed special characters within the document. One special character may turn on boldface, while another selects italics. Although these special characters are meaningful to the word processor, the characters confuse TYPE, resulting in unrecognizable output. In other words, if you create a file with your word processor, you'll have to use the word processor to view the file's contents.

The TYPE command displays unformatted text files, which are also called *ASCII files*. Examples of text files include batch files (with the BAT extension) that you will create in Chapter 15.

If TYPE is unable to locate the file you specify, TYPE will display an error message like this:

```
C:\> TYPE  FILENAME.EXT
File not found - FILENAME.EXT

C:\>
```

If this error message appears, make sure you are spelling the filename correctly and that you are specifying the correct disk drive letter or directory name. If the file of interest resides on a different disk, precede the filename with a disk drive letter and colon. The following TYPE command, for example, would display the contents of a file called FILENAME.EXT on the disk in drive A.

```
C:\> TYPE  A:FILENAME.EXT <ENTER>
```

COPY Duplicates a File's Contents

In an office, it is common to make copies of your paper files, either as backup copies or to distribute to other coworkers. These same scenarios hold true for the files you store on disk. To help you duplicate your disk files, DOS provides the COPY command. First, format an unused floppy disk as discussed in Chapter 7. Next, place the floppy disk in drive A. Using the COPY command, you will copy files from your hard disk to this floppy.

The COPY command copies the contents of a file in one location to another. To use COPY, therefore, you must specify the name of the file you want to duplicate and where you want the file copy to go. As you learned in Chapter 7, the FORMAT command is an external command stored in the file FORMAT.COM. Like all your external DOS commands, FORMAT.COM is in the DOS directory. The following DIR command displays the directory entry for FORMAT.COM:

```
C:\> DIR  \DOS\FORMAT.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

FORMAT   COM     22717 02-12-93   6:00a
        1 file(s)     22717 bytes
                   16762880 bytes free
```

The following COPY command copies the contents of the file FORMAT.COM from the DOS directory of the current drive (in this case, C) to the floppy disk in drive A:

```
C:\> COPY  \DOS\FORMAT.COM  A:FORMAT.COM <ENTER>
     1 File(s) copied
```

When you execute this COPY command, you will see your hard disk's drive activation light illuminate briefly as DOS reads the file's contents. Next, drive A's activation light will illuminate as DOS copies the file's contents to the disk. When COPY successfully completes, COPY will display a message indicating the number of files copied. Using the following DIR command, you can verify that the COPY operation was successful by viewing the file FORMAT.COM in the directory listing for drive A.

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
        1 file(s)      22717 bytes
                     1190912 bytes free
```

If COPY is unable to locate the file specified, COPY will display an error message indicating it did not find the file:

```
C:\> COPY  FILENAME.EXT  A:FILENAME.EXT <ENTER>
File not found
     0 File(s) copied

C:\>
```

If this error message appears, double-check the spelling of your filename. Also, make sure you are specifying the correct directory name for the file if necessary.

As shown in Figure 8-1, file copy operations specify the names of a *source file* to copy, and a *destination* or *target file* to which the copy is made.

In the previous examples, the source and destination filenames were the same. When you copy files, COPY lets you use any destination name you like. For example, the following COPY command again copies the contents of the file FORMAT.COM to drive A. This time, however, the destination filename is FMAT.COM.

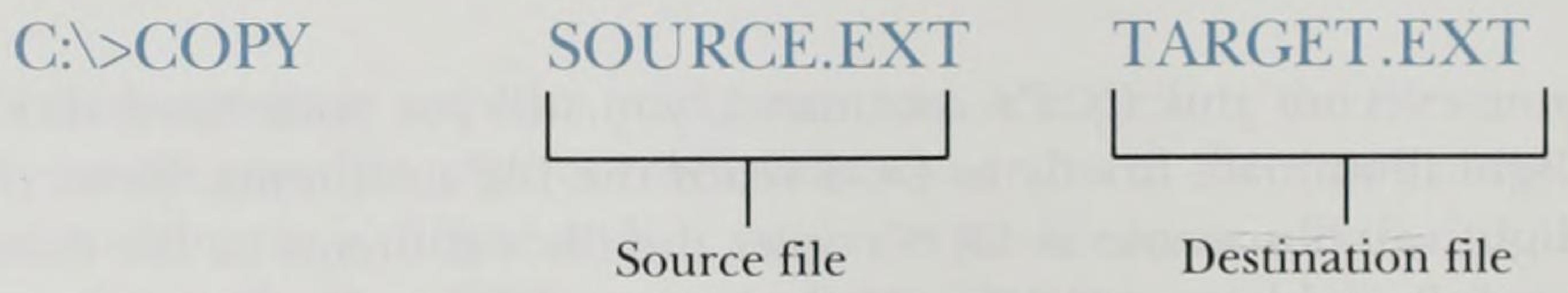


Figure 8-1. *Each file copy operation has a source and destination filename*

```
C:\> COPY  \DOS\FORMAT.COM  A:FMAT.COM <ENTER>
        1 File(s) copied
```

A directory listing of drive A reveals entries for both copies of the file:

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
FMAT     COM     22717 02-12-93   6:00a
        2 file(s)      45434 bytes
                     1167872 bytes free
```

If you specify only a drive letter for the destination of a file copy, COPY will use the source file's name. The following COPY command copies the contents of the file DISKCOPY.COM from the DOS directory to the disk in drive A:

```
C:\> COPY  \DOS\DISKCOPY.COM  A: <ENTER>
        1 File(s) copied
```

Because the destination did not include a filename, COPY used the source name, as shown in Figure 8-2.

A directory listing of drive A reveals the file DISKCOPY.COM:

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
FMAT     COM     22717 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
        3 file(s)      57313 bytes
                     1155584 bytes free
```

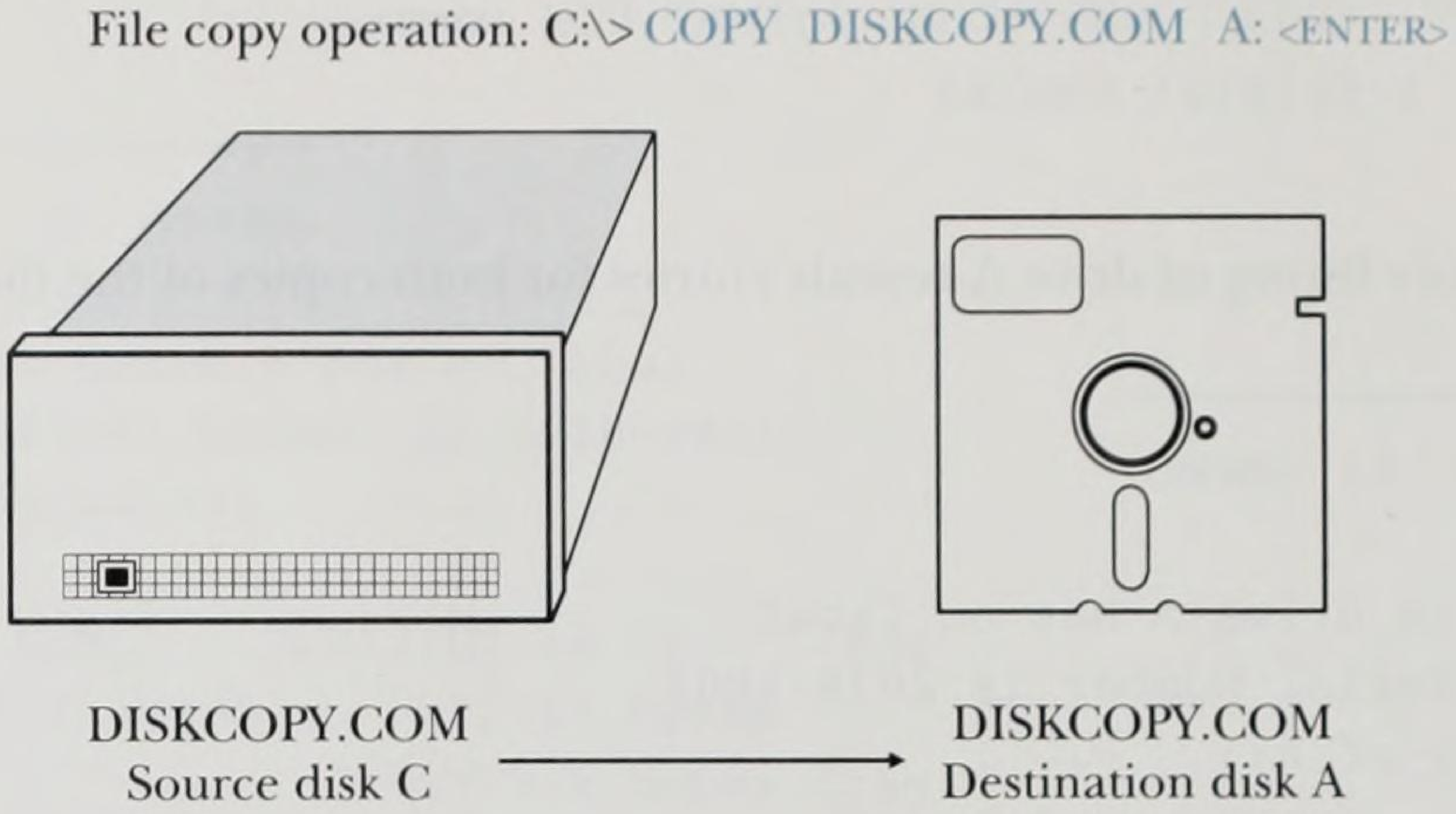


Figure 8-2.. *If the destination of a file copy operation does not specify a filename, COPY will use the source name*

WARNING

COPY is a destructive command. This means it will overwrite the contents of a file whose name matches the destination filename of a COPY operation.

To demonstrate how COPY can overwrite a file, consider the following command that copies the file DISKCOPY.COM to drive A. The destination filename used is FMAT.COM.

```
C:\>       \DOS\DISKCOPY.COM  A:FMAT.COM <ENTER>
     1 File(s) copied
```

In this case, COPY performed the operation even though a file already existed with the same name as the destination file. The COPY command overwrote the previous file's contents. If you perform a directory listing of drive A, you will find that the file size of FMAT.COM now matches the size of DISKCOPY.COM, indicating that FMAT.COM was indeed overwritten.

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
```

```
FMAT     COM      11879 02-12-93   6:00a
DISKCOPY COM      11879 02-12-93   6:00a
       3 file(s)        46475 bytes
                      1166336 bytes free
```

In later chapters you will learn how to protect your files from being inadvertently deleted or overwritten in this way. For now, however, use the DIR command to determine if a file exists with the same name as the proposed destination file. If a file with the same name exists, choose a different destination name.

RENAME Changes a File's Name

Just as you must change the name of a paper file in your office when the name no longer reflects the file's contents, so it is true for your disk files. The RENAME command lets you rename an existing file. Because RENAME is used so frequently, DOS lets you abbreviate RENAME as REN. In order to rename the files you just copied to the floppy disk in drive A, select drive A as the current drive:

```
C:\> A: <ENTER>

A:\>
```

The format of the RENAME command is

RENAME *OLDNAME NEWNAME*

The following RENAME command renames the file FMAT.COM as DCOPY.COM.

```
A:\> RENAME  FMAT.COM  DCOPY.COM <ENTER>

A:\>
```

If RENAME is successful, it completes, and DOS redisplays its prompt. RENAME does not display a message indicating success. You can verify that the rename operation was successful using DIR:

```
A:\> DIR <ENTER>
 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
DCOPY    COM     11879 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
        3 file(s)      46475 bytes
                     1166336 bytes free
```

Do not confuse renaming a file with moving a file. RENAME will only change the name that appears in a directory listing. RENAME will not move a file from one drive or directory to another. If you try to rename a file from one disk to another, RENAME will display an error message, as shown here:

```
A> RENAME  A:DISKCOPY.COM  C: <ENTER>
Invalid parameter
```

Also, RENAME will not let you rename a file to a name that is already in use. If you try to do so, RENAME will display an error message telling you that a duplicate name exists in the directory:

```
A:\> RENAME  DISKCOPY.COM  FORMAT.COM <ENTER>
Duplicate file name or file not found
```

If RENAME cannot locate the file you want to rename, RENAME will display this same message, indicating it did not find the file.

NOTE *Do not rename any file on your disk until you are sure you understand the file's contents and use. Many of your applications programs expect to find files with specific names. If you rename such a file, the application may fail.*

DEL Deletes Files

In an office you can simply throw a file away when it is no longer needed. For files on your disk, you can delete files so DOS can reuse the disk space that previously held the file. The DEL command lets you delete a file from your disk. For example, the following command deletes the file DCOPY.COM from the current drive (in this case, drive A):

```
A:\> DEL  DCOPY.COM <ENTER>

A:\>
```

Like RENAME, the DEL command does not display a message indicating its success. In this case, you can use DIR to verify the file DCOPY.COM has been deleted.

```
A:\> DIR <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

FORMAT   COM     22717 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
        2 file(s)      34596 bytes
                     1178624 bytes free
```

Note that the amount of available disk space increases when you delete a file.

NOTE

If you inadvertently delete the wrong file, there are third-party software packages you can use to retrieve the file's contents. If you need to undelete a file in this way, do not copy any files to the disk until you have successfully retrieved the file. If you are using DOS 5 or later, you can use the UNDELETE command discussed in Chapter 28.

If DEL cannot locate the file specified, DEL will display a "File not found" message:

```
A:\> DEL  FILENAME.EXT <ENTER>
File not found
```

If the file you want to delete is on a different disk, precede the filename with the correct drive letter and a colon.

```
C:\> DEL  A:FILENAME.EXT <ENTER>
```

NOTE

Do not delete files from your disk whose contents or purpose you don't fully understand. If you delete a critical file, you may prevent an application from running, or even prevent DOS from being able to start.

SUMMARY

The TYPE Command

The TYPE command lets you display a text file's contents on your screen. Text files contain only standard letters and numbers. The format of the TYPE command is as follows:

```
C:\> TYPE FILENAME.EXT <ENTER>
```

The COPY Command

The COPY command lets you duplicate a file's contents by copying the file to a new location. File copy operations refer to the file you are copying as the source file. Likewise, the location you are copying the file to is the destination. The format of the COPY command is as follows:

```
C:\> COPY SOURCE DESTINATION <ENTER>
```

The RENAME Command

The RENAME command lets you change the name of an existing file. Because of its frequency of use, DOS lets you abbreviate RENAME as REN. The RENAME command only renames a file within a directory, RENAME cannot move a file from one disk or directory to another. The format of the RENAME command is as follows:

```
C:\> RENAME OLDNAME  NEWNAME <ENTER>
```

The DEL Command

The DEL command lets you remove files from your disk when the files are no longer needed. When deleting a file, DOS makes available for reuse the disk space that the file contained. Should you inadvertently delete the wrong file, you may be able to "undelete" it by using a third-party software package or the DOS 5 or later UNDELETE command. The format of the DEL command is

```
C:\> DEL FILENAME.EXT <ENTER>
```

Hands On

Using the following COPY command, copy the file DISKCOPY.COM from the DOS directory to the floppy disk in drive A. If drive C is not the current drive, select drive C before you continue.

```
C:\> COPY  \DOS\DISKCOPY.COM  A: <ENTER>
         1 File(s) copied
```

Use the DIR command to verify that the operation was successful.

```
C:\> DIR  A: <ENTER>
```

Next, repeat the COPY command, but this time, name the target file DC.COM.

```
C:\> COPY  \DOS\DISKCOPY.COM  A:DC.COM <ENTER>
```

Next, select drive A as the default drive.

```
C:\> A: <ENTER>

A:\>
```

Now RENAME DC.COM as DCOPY.COM, using the command abbreviation REN.

```
A:\> REN  DC.COM  DCOPY.COM <ENTER>
```

Use DIR to verify that the rename operation was successful.

```
A:\> DIR <ENTER>
```

Note the amount of available space on your disk. Using DEL, delete the file DISKCOPY.COM.

```
A:\> DEL  DISKCOPY.COM <ENTER>
```

Using DIR, verify the file was deleted and that DOS has made the file's disk space available for reuse.

Last, use the following TYPE command to display the contents of the file CONFIG.SYS that resides on drive C.

```
A:\> TYPE  C:\CONFIG.SYS <ENTER>
```

If TYPE displays the "File not found" message, don't worry; you will create a CONFIG.SYS file in Chapter 24.

Review

1. When can using the COPY command result in the loss of information from your disk?
2. What filename does COPY use for the destination in the following COPY command, and why?

   ```
   C:\> COPY \DOS\FDISK.EXE A: <ENTER>
   ```

3. Why will the following command fail?

   ```
   C:\> REN \DOS\FORMAT.COM A: <ENTER>
   Invalid Parameter
   ```

4. What is a text file?
5. How do the following commands differ?

   ```
   C:\> RENAME OLDNAME.EXT NEWNAME.EXT <ENTER>

   C:\> REN OLDNAME.EXT NEWNAME.EXT <ENTER>
   ```

Answers

1. COPY is a destructive command, meaning COPY overwrites the contents of a file with the same name as the destination file when specified in a COPY command. For example, given the following COPY command,

   ```
   C:\> COPY  SOURCE.EXT  DESTIN.EXT <ENTER>
   ```

   COPY will overwrite the contents of a file named DESTIN.EXT if such a file exists in the current directory. After COPY overwrites the file's contents, the information previously contained in the file is lost.

2. If you only specify a target drive letter as the destination of a file copy operation, COPY will duplicate the source filename for the destination file. In this case, COPY will create a file on drive A named FDISK.EXE.

3. The RENAME command will not move a file from one disk or directory to another. In this case, RENAME is used incorrectly in an attempt to move the file FORMAT.COM from the DOS directory on drive C to the disk in drive A. RENAME cannot perform such processing, and the command will fail.

4. A text file (sometimes called an ASCII file) is a file containing only standard letters and numbers. Examples of text files include DOS batch files. The TYPE command lets you view a text file's contents on your screen. Word processing documents and external command files contain nonstandard characters that are meaningless to TYPE, and will appear on your screen as unrecognizable characters.

5. The commands are the same. Due to RENAME's frequency of use, DOS lets you abbreviate the RENAME command as REN.

Advanced Concepts

Using DOS *wildcard characters* (characters you can substitute for a single or group of characters in a filename), you can delete several files at one time. If you are using DOS 4 or later, you can include the /P switch in DEL's command line as shown here:

```
C:\> DEL *.EXT /P <ENTER>
```

The /P switch directs DEL to individually display each filename and prompt you to determine if you want to delete the file. When you use /P, DEL will display the following prompt for each file.

```
FILENAME.EXT,  Delete (Y/N)?
```

To delete the file, type **Y** and press ENTER. To leave the file on disk, type **N** and press ENTER. Using the /P switch, DEL lets you selectively delete files.

An ASCII or text file is a file that contains standard characters and numbers only. Depending on your application, there may be times when you need to combine two or more text files. For example, assume you have three files named START, MIDDLE, and END that contain the contents shown in Figure 8-3

By placing the plus sign between filenames, COPY lets you combine all three of these files into one file named WHOLE, as shown here.

```
C:\> COPY  START+MIDDLE+END  WHOLE <ENTER>
```

In this case, WHOLE will contain the following:

```
C:\> TYPE  WHOLE <ENTER>
UNLV       UNLV       UNLV
RUNNIN     RUNNIN     RUNNIN
REBELS!    REBELS!    REBELS!
```

The COPY command even lets you use wildcard characters in such operations. For example, the following command copies all the files with the DAT extension and then appends files with the TXT extension to create a file called LARGE.

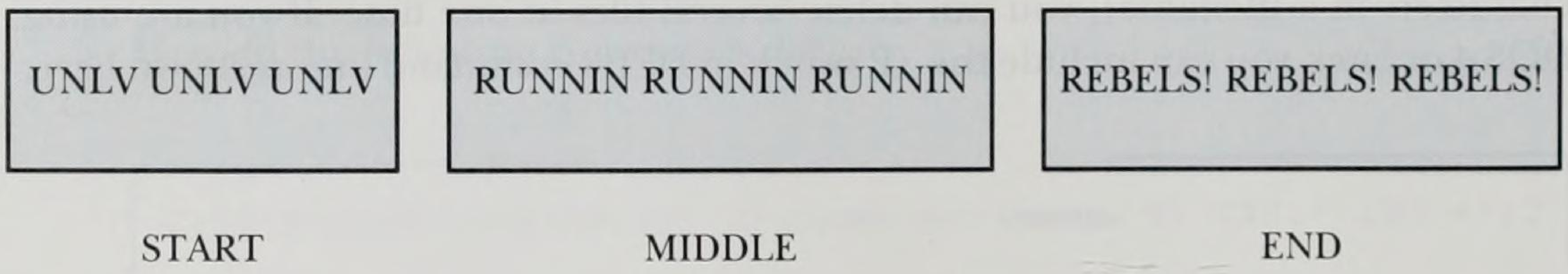


Figure 8-3. *The contents of the files START, MIDDLE, and END*

```
C:\> COPY  *.DAT+*.TXT  LARGE <ENTER>
```

Make sure you don't specify the name of the destination file as the last filename to copy. If you do, COPY tries to copy the file into itself and fails:

```
C:\> COPY  ONE+TWO  TWO <ENTER>
Contents of destination lost before copy
```

Many users ask when to use the plus sign in a file copy operation. The most common case is to add a line or two to a batch file. For example, assume you want to add the CLS ("clear the screen") command to the end of your AUTOEXEC.BAT file. Rather than spending time invoking an editor or word processor, you can quickly append a line by copying from the CON device (console) as shown here:

```
C:\> COPY  AUTOEXEC.BAT+CON  AUTOEXEC.BAT <ENTER>
```

COPY will first copy AUTOEXEC.BAT's current contents, displaying the file's name. Next, DOS will display the word CON, indicating it is ready for you to type from the keyboard. In this case, type **CLS** and press F6 followed by ENTER:

```
C:\> COPY  AUTOEXEC.BAT+CON  AUTOEXEC.BAT <ENTER>
AUTOEXEC.BAT
CON
CLS <ENTER>
^Z <ENTER>
        1 File(s) copied

C:\>
```

Abbreviating Directory Commands

Using the *.* wildcard combination, you can reference all of the files in a directory. For example, the following command displays a directory listing for the directory SUBDIR:

```
C:\> DIR  \SUBDIR\*.* <ENTER>
```

However, by default, if you omit the wildcards, DIR will still display a directory listing of all the files in the directory.

```
C:\> DIR  \SUBDIR <ENTER>
```

Most DOS commands default to the *.* wildcard combination if you specify only a directory. For example, the following COPY command will copy all of the files in the directory SUBDIR to drive A:

```
C:\> COPY  \SUBDIR  A: <ENTER>
```

Likewise, the following DEL command will delete all of the files in SUBDIR:

```
C:\> DEL  \SUBDIR <ENTER>
All files in directory will be deleted!
Are you sure (Y/N)?
```

By letting you abbreviate subdirectory names in this way, DOS saves you time and keystrokes.

Key Terms

Text file A file containing only standard characters and numbers, as opposed to embedded word processor characters or computer instructions. Text files are often called ASCII files.

Source file The file whose contents a command will copy, display, rename, or delete.

Destination file The file to which a source file's contents are being written or renamed.

Chapter 9

DOS Wildcard Characters

In Chapter 8, you learned how to copy, delete, rename, and display files. All the commands you issued in Chapter 8 worked with one file at a time. You can save time and typing in many cases by working with groups of files, for example, all the files with the LTR extension in a directory. In this chapter, you will learn how to perform file operations on groups of related files using the DOS *wildcard characters,* * and ?. DOS wildcard characters are so named because like "wildcards" in a card game, which you can use as cards of your choice, wildcards in filenames let DOS substitute letters to match groups of files. DOS wildcards quickly expand the number of files a command can manipulate at one time.

The * Wildcard

Wildcards help you match groups of files with similar names. In theory, wildcards let you specify character positions that DOS can ignore when trying to match filenames. The *asterisk wildcard* (*) is the most powerful and most commonly used wildcard. When you place the asterisk in a filename, you tell DOS that you don't care about matching the characters that appear in the wildcard's position, as well as those that follow. For example, the following DIR command, at the top of the next page, uses the asterisk to list all the files in the DOS directory that have the COM extension.

```
C:\> DIR  \DOS\*.COM <ENTER>
```

In DOS 6, the command will display the following:

```
C:\> DIR \DOS\*.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

FORMAT   COM     22717 02-12-93   6:00a
KEYB     COM     14983 02-12-93   6:00a
EDIT     COM       413 02-12-93   6:00a
MORE     COM      2546 02-12-93   6:00a
SYS      COM      9379 02-12-93   6:00a
UNFORMAT COM     12738 02-12-93   6:00a
CHOICE   COM      1754 02-12-93   6:00a
DOSSHELL COM      4620 02-12-93   6:00a
HELP     COM       413 02-12-93   6:00a
MODE     COM     23521 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
GRAPHICS COM     19694 02-12-93   6:00a
LOADFIX  COM      1131 02-12-93   6:00a
TREE     COM      6898 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
MOUSE    COM     56408 02-12-93   6:00a
VSAFE    COM     62576 02-12-93   6:00a
COMMAND  COM     52925 02-12-93   6:00a
       19 file(s)     321098 bytes
                    16762880 bytes free
```

The asterisk directs DOS to ignore the character position where the wildcard appears, as well as all the character positions that follow. In this case, because the asterisk appears in the base name's first character position, DOS ignores the entire base name, matching all files that have the COM extension.

When you use a wildcard character, DOS examines every file in the directory to determine if the file matches the wildcard. If the file matches, DOS uses the file entry. If the file does not match, DOS continues its search with the next file.

Using the same technique, the following command lists all the files in the DOS directory with the EXE extension:

```
C:\> DIR  \DOS\*.EXE <ENTER>
```

By placing an asterisk in the second character position of the base name and one in the extension, you can list all the files in the DOS directory that begin with the letter D.

```
C:\> DIR  \DOS\D*.* <ENTER>
```

In DOS 6, the command will display the following:

```
C:\> DIR \DOS\D*.* <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

DBLSPACE BIN     50284 02-12-93   6:00a
DEBUG    EXE     15715 02-12-93   6:00a
DOSSHELL VID      9462 02-12-93   6:00a
DOSSHELL INI     16912 02-13-93   7:49p
DOSSHELL GRB      4421 02-12-93   6:00a
DEFRAG   EXE     75017 02-12-93   6:00a
  :        :        :        :        :
DOSKEY   COM      5883 02-12-93   6:00a
DBLSPACE EXE    273068 02-12-93   6:00a
DBLSPACE HLP     72173 02-12-93   6:00a
DBLSPACE INF      2166 02-12-93   6:00a
DBLSPACE SYS       339 02-12-93   6:00a
DBLWIN   HLP      8597 02-12-93   6:00a
DOSSHELL HLP    161323 02-12-93   6:00a
DEFAULT  SET      4482 02-08-93   7:51p
DEFAULT  SLT        21 02-08-93   1:58p
DEFAULT  CAT        66 02-08-93   7:51p
       26 file(s)    1028836 bytes
                    16762880 bytes free
```

In this case, the wildcard combination tells DOS you only care about the first letter of the file's base name.

If you replace the asterisk in the extension with COM, DIR will display the following:

```
C:\> DIR \DOS\D*.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
        4 file(s)      33002 bytes
                    16762880 bytes free
```

Remember, once DOS encounters the asterisk within a filename, it ignores all the characters that follow. For example, consider the following DIR command:

```
C:\> DIR  \DOS\D*BUG.COM <ENTER>
```

Because the asterisk wildcard appears in the second character position, DOS ignores all the characters that follow (in this case, DOS ignores "BUG"). As a result, DOS displays

```
C:\> DIR \DOS\D*BUG.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
        4 file(s)      33002 bytes
                    16762880 bytes free
```

Last, the following command displays all the files in the DOS directory:

```
C:\> DIR  \DOS\*.* <ENTER>
```

The wildcard combination, in this case, tells DOS to ignore both the base name and extension. DIR, by default, does this anyway. As a result, the following DIR command is equivalent:

```
C:\> DIR  \DOS <ENTER>
```

Using the * for Other Operations

The previous examples used the asterisk wildcard with DIR to illustrate how DOS uses the wildcard to let you select groups of files. Most of the DOS file-manipulation commands you will use throughout this book support the DOS wildcards. In this section you will use the asterisk wildcard with COPY, RENAME, and DEL.

To begin, place a formatted blank disk in drive A. Next, using the following COPY command, copy to the disk in drive A all the DOS commands that begin with the letter D and have the COM extension.

```
C:\> COPY  \DOS\D*.COM  A: <ENTER>
```

In this case, COPY will copy each matching file to drive A, displaying the file's name as it begins the copy operation:

```
C:\> COPY  \DOS\D*.COM  A: <ENTER>
C:\DOS\DOSKEY.COM
C:\DOS\DOSSHELL.COM
C:\DOS\DISKCOMP.COM
C:\DOS\DISKCOPY.COM
        4 file(s) copied
```

As discussed in Chapter 8, if you don't specify a destination filename, COPY uses the source filename. A directory listing of drive A, as shown on the next page, reveals that DOS has copied the files as desired.

```
C:\> DIR A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
        4 file(s)      33002 bytes
                     1179648 bytes free
```

In a similar way, this command copies to drive A all the files in the DOS directory that begin with the letter A.

```
C:\> COPY \DOS\A*.* A: <ENTER>
ANSI.SYS
ATTRIB.EXE
APPEND.EXE
        3 file(s) copied
```

The disk in drive A now contains the following:

```
C:\> DIR A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
ANSI     SYS      9065 02-12-93   6:00a
ATTRIB   EXE     11165 02-12-93   6:00a
APPEND   EXE     10774 02-12-93   6:00a
        7 file(s)      64006 bytes
                     1147904 bytes free
```

Disks, like cassette or videotapes, do not have an unlimited amount of space. If, for example, you try to copy all the files from the DOS directory to the disk in drive A, the floppy disk will eventually run out of space. As a result, DOS will display the following error message:

```
Insufficient disk space
```

Should this error message occur, you can use DIR to determine the last file successfully copied and resume the copy operation with a new floppy disk. As you will learn in Chapter 23, the COPY command really doesn't give you an easy way to resume the copy operation at the last file copied. A better way to copy a large number of files to many floppy disks is to use the XCOPY command, presented in Chapter 23.

Select drive A as the default drive.

```
C:\> A: <ENTER>

A:\>
```

Next, using the following RENAME command, rename all the files beginning with the word "DISK" (such as DISKCOPY and DISKCOMP) by replacing "DISK" with the letters "FLOP" (yielding FLOPCOPY and FLOPCOMP).

```
A:\> REN  DISK*.*  FLOP*.* <ENTER>
```

A directory listing of drive A reveals that REN has successfully renamed the files:

```
A:\> DIR <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

DOSSHELL COM       4620 02-12-93    6:00a
FLOPCOMP COM      10620 02-12-93    6:00a
FLOPCOPY COM      11879 02-12-93    6:00a
DOSKEY   COM       5883 02-12-93    6:00a
ANSI     SYS       9065 02-12-93    6:00a
```

```
ATTRIB   EXE    11165 02-12-93   6:00a
APPEND   EXE    10774 02-12-93   6:00a
        7 file(s)     64006 bytes
                    1147904 bytes free
```

Likewise, the following command renames the file ANSI.SYS to ANSIFILE.SYS:

```
A:\> REN ANSI.* ANSIFILE.* <ENTER>
```

In this case, the wildcard character tells DOS to leave the extension unchanged.

```
A:\> DIR <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

DOSSHELL COM      4620 02-12-93   6:00a
FLOPCOMP COM     10620 02-12-93   6:00a
FLOPCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
ANSIFILE SYS      9065 02-12-93   6:00a
ATTRIB   EXE     11165 02-12-93   6:00a
APPEND   EXE     10774 02-12-93   6:00a
        7 file(s)     64006 bytes
                    1147904 bytes free
```

You need to be careful using the DEL command with DOS wildcards. If you specify the wrong wildcard combination, you can inadvertently delete needed files from your disk. To reduce the possibility of inadvertent file deletions, you may want to first issue a DIR command using the wildcard combination you plan to use with DEL. By using DIR first, you can verify that DIR lists the files you really want to delete and no others.

The following command deletes all the files with the COM extension from the floppy disk. Make sure drive A is the default drive before you execute this command.

```
A:\> DEL *.COM <ENTER>
```

A directory listing of drive A reveals DEL has successfully deleted the files.

```
A:\> DIR <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 203B-1E03
 Directory of A:\

ANSIFILE SYS       9065 02-12-93    6:00a
ATTRIB   EXE      11165 02-12-93    6:00a
APPEND   EXE      10774 02-12-93    6:00a
        3 file(s)       31004 bytes
                      1182208 bytes free
```

Because using the wildcard combination *.* with DEL can have a devastating effect on your files if you invoke it in error, DEL will first prompt you to verify you really want to delete all the files in the directory when you use *.* as shown here:

```
A:\> DEL *.* <ENTER>
All files in directory will be deleted!
Are you sure (Y/N)?
```

If you are sure you are deleting the correct files, type Y and press ENTER, and DEL will delete the files. If you aren't sure, type N and press ENTER. Use DIR to verify the correct files will be deleted.

The ? Wildcard

As discussed, when DOS encounters the asterisk wildcard, it ignores the characters in the wildcard's position and all characters that follow. The *question mark* (?) *wildcard* gives you finer control over character matching. The question mark wildcard provides a one-for-one match. In other words, when DOS encounters the ? wildcard, DOS ignores only the character appearing in the same position as the wildcard. DOS still considers all the characters that follow the wildcard.

The question mark wildcard is less frequently used than the asterisk. The following examples illustrate its use. To begin, the following DIR command displays COM files in a DOS directory that begin with the letters "DISKCO."

```
C:\> DIR \DOS\DISKCO?? <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
        2 file(s)      22499 bytes
                    16762880 bytes free
```

In this case, the two question mark wildcards direct DIR to ignore the last two characters of the filename. Note that the DIR command does not include a file extension. By default, if you don't specify a file extension, DIR uses * to match all extensions.

This command tries to display the same two files, using only the letters "DISK" and a single question mark.

```
C:\> DIR  \DOS\DISK? <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\DOS

File not found
```

In this case, DOS displays the "File not found" message because the directory does not contain any five-letter filenames that begin with the letters "DISK." Remember, the question mark wildcard only replaces a single character.

Last, this command displays the files in the DOS directory whose base names are only four characters or less.

```
C:\> DIR  \DOS\???? <ENTER>
```

In DOS 6, the command will display

```
C:\> DIR \DOS\???? <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

.            <DIR>       11-23-92   9:26p
..           <DIR>       11-23-92   9:26p
KEYB     COM      14983 02-12-93   6:00a
ANSI     SYS       9065 02-12-93   6:00a
EDIT     COM        413 02-12-93   6:00a
MORE     COM       2546 02-12-93   6:00a
MSD      EXE     158470 02-12-93   6:00a
SYS      COM       9379 02-12-93   6:00a
EGA      CPI      58870 02-12-93   6:00a
EGA      SYS       4885 02-12-93   6:00a
MEM      EXE      32150 02-12-93   6:00a
MOVE     EXE      17371 02-12-93   6:00a
EDIT     HLP      17898 02-12-93   6:00a
HELP     HLP     294741 02-12-93   6:00a
HELP     COM        413 02-12-93   6:00a
MODE     COM      23521 02-12-93   6:00a
FC       EXE      18650 02-12-93   6:00a
FIND     EXE       6770 02-12-93   6:00a
SORT     EXE       6922 02-12-93   6:00a
TREE     COM       6898 02-12-93   6:00a
MSAV     EXE     172198 02-12-93   6:00a
MSAV     HLP      23891 02-12-93   6:00a
MWAV     EXE     142640 02-12-93   6:00a
MWAV     HLP      25663 02-12-93   6:00a
       24 file(s)    1048337 bytes
                    16762880 bytes free
```

The question mark results in a one-for-one match. In this case, a single letter matches, two letters match, three letters match, and finally, four letters match.

SUMMARY

* Wildcard

The asterisk wildcard helps you specify "don't care" character positions in filenames so that DOS will match groups of files. When DOS encounters the asterisk, DOS ignores the characters in the position containing the wildcard, as well as all the characters that follow it. The following command, for example, lists all the files in a directory with the BAT extension.

```
C:\>DIR *.BAT <ENTER>
```

? Wildcard

The question mark wildcard lets you specify a single "don't care" position in file operations. When DOS encounters the question mark, DOS ignores only the character in the position containing the question mark. The following command, for example, lists files beginning with the letters DISKCO.

```
C:\>DIR DISKCO?? <ENTER>
```

Hands On

Using the DIR command, list all files in the DOS directory with the EXE extension.

```
C:\> DIR  \DOS\*.EXE <ENTER>
```

Next, change the command slightly to display the files with the SYS extension.

```
C:\> DIR  \DOS\*.SYS <ENTER>
```

Next, list the files that begin with C.

```
C:\> DIR  \DOS\C* <ENTER>
```

By changing the command, you can display only files with the COM extension that begin with C.

```
C:\> DIR  \DOS\C*.COM <ENTER>
```

Place an unused formatted disk in drive A. Next, issue the following command to copy all the files with the SYS extension from the DOS directory to drive A.

```
C:\> COPY  \DOS\*.SYS  A: <ENTER>
```

Use the following DIR command to verify that the copy operation was successful.

```
C:\> DIR  A: <ENTER>
```

Next, use the following command to copy the files from the DOS directory that begin with F to drive A.

```
C:\> COPY  \DOS\F*.*  A: <ENTER>
```

Select drive A as the current drive.

```
C:\> A: <ENTER>

A:\>
```

Use the following RENAME command to rename the file FDISK.COM to F.COM.

```
A:\> REN FDISK.* F.* <ENTER>
```

Use DIR to verify that the rename operation was successful.

```
A:\> DIR <ENTER>
```

Next, delete all the files with the SYS extension.

```
A:\> DEL  *.SYS <ENTER>
```

Using this same technique, delete all the files from the floppy disk.

```
A:\> DEL  *.* <ENTER>
```

To prevent you from inadvertently deleting all the files in the directory, DEL will first prompt you with the following:

```
All files in directory will be deleted!
Are you sure (Y/N)?
```

To delete the files, type Y and press ENTER.

Review

1. What is a wildcard?
2. What command would list files in the DOS directory that begin with B?
3. What is the command to copy all the files with the SYS extension from the DOS directory to a disk in drive A?
4. How does the asterisk wildcard (*) differ from the question mark (?).
5. What is the function of the following command?

```
A:\> REN  FILENAME.*  F.*  <ENTER>
```

Answers

1. A wildcard is a symbol you can place within a DOS filename that directs DOS to ignore the symbol's position, and depending on the wildcard, the character positions that follow, to match groups of similarly named files. DOS supports the * and ? wildcards. Wildcards exist to help you perform file-manipulation commands on groups of files.
2. The command to list the files in the DOS directory that begin with B is

```
C:\> DIR  \DOS\B*  <ENTER>
```

3. The command to copy all the files with the SYS extension from the DOS directory to a disk in drive A is

```
C:\> COPY  \DOS\*.SYS  A:  <ENTER>
```

4. The asterisk wildcard (*) directs DOS to not only ignore the character in the wildcard's position, but also the characters in all the positions following the wildcard. The question mark wildcard (?), on the other hand, directs DOS to ignore filename character positions on a one-for-one basis.
5. The command renames every file whose base name is FILENAME to the base name F, regardless of the file's extension.

Advanced Concepts

Most of the DOS commands support wildcard characters. For those that don't, you can create a batch file named WILD.BAT that lets you add wildcard support.

```
@ECHO OFF
REM WILD.BAT
REM Written By: Kris Jamsa
REM Function: Lets commands that aren't written to support
REM wildcard characters behave as if they were.
REM Format: WILD CommandName Parameters
REM Example: WILD TYPE *.BAT *.SYS

IF '%1'=='' GOTO NO_COMMAND
```

```
REM Save the desired command in the named parameter COMMAND
SET COMMAND=%1

REM Loop through the desired file names
:LOOP
  SHIFT
  FOR %%I IN (%1) DO %COMMAND% %%I
IF NOT ’%1’==’’ GOTO LOOP

REM No more files remain
GOTO DONE
:NO_COMMAND
ECHO WILD: Must specify desired command
:DONE
```

To use wildcards with the TYPE command, for example, you can invoke WILD as shown here:

```
C:\> WILD TYPE *.BAT <ENTER>
```

In addition to supporting wildcards, this batch file lets you place several filenames in the command line, as shown here:

```
C:\> WILD TYPE \AUTOEXEC.BAT \CONFIG.SYS <ENTER>
```

Because the batch file uses named parameters, it requires DOS version 3.3 or later. The batch file uses the FOR command to provide wildcard support and the SHIFT command to access multiple files.

If you don’t want to create the batch file, or if you are not using DOS version 3.3 or higher, you can also combine the FOR command with TYPE from the command line, as shown here:

```
C:\> FOR %I IN (*.BAT) DO TYPE %I <ENTER>
```

In this case, FOR behaves as it would within a batch file. Note, however, that when you invoke FOR from the DOS prompt, you only include one percent (%) sign before the variable name. The FOR command is discussed in detail in Chapter 17.

Key Term

Wildcard A character or symbol that DOS uses in matching groups of files. DOS supports the asterisk (*) and question mark (?) wildcards.

Chapter 10

Directories

In Chapter 4, you found that DOS helps you organize files on your disk by grouping related files into lists called directories. As you learned to work with DIR and COPY, you made extensive use of files in the DOS directory. Almost every software program you buy will eventually have its own directory. In most cases, the installation program that places the program into your hard disk will create the directory for you. As you create your own files, however, it won't take long before you will want to organize them by creating your own directories. In this chapter you will learn how to create a directory using MKDIR, how to select a specific directory as the default using CHDIR, and how to use RMDIR to remove a directory when it is no longer needed.

The Root Directory

When you first format a disk, DOS creates a single directory on your disk called the *root directory.* The root directory is so named because all the directories you later create on your disk grow from the root, much like the branches of a tree. Conceptually, the root directory is much like a filing cabinet—you can put lots of files loose in the cabinet, or you can create drawers (directories) or even smaller drawers (called subdirectories) with different names to put your files in to organize them. When you create a directory in the root to store specific files, you essentially label one of the filing cabinet drawers.

DOS uses the backslash character (\) to represent "directory" or "subdirectory." The backslash alone represents the root directory—since the root is the "home" directory, DOS considers it as the default to be referred to when no name is specified immediately after the backslash. Using this symbol, the following DIR command lists the files and directories that reside in the root:

```
C:\> DIR  \ <ENTER>
```

Make sure you use the backslash character (\), and *not* the forward slash (/). Depending on the different application programs on your disk, the number and names of directories in the root will differ. At a minimum, you will probably find a directory entry named DOS. In addition, you will probably see file entries for COMMAND.COM, CONFIG.SYS, and AUTOEXEC.BAT. As you will learn, DOS uses these three files each time DOS starts. As a rule, these are the only files you should have in the root directory. In other words, the root "filing cabinet" should contain primarily directories (drawers), not big piles of loose folders (files).

The Current Directory

You have learned that DOS lets you select one disk drive as the current or default drive that, unless you specify otherwise, it searches for external commands and files. In the same way, DOS lets you select one of your directories as the *current directory*, in which DOS searches unless you tell it otherwise.

Conceptually, selecting the current directory is equivalent to closing one drawer of a filing cabinet and opening another—when you ask for a file, DOS looks in the open drawer. By default, when your computer starts, the root directory is the current directory. To change the current directory, you must use the CHDIR command.

Tracking the Current Directory with the DOS Prompt

As you have learned, when you change the current drive, DOS changes its prompt to include the current drive letter. For example, watch how the prompt changes as you change the default drive from C to A.

```
C> A: <ENTER>

A>
```

To help you keep track of the directory you are currently using, you can direct DOS to include the current directory as well as the current drive in the prompt. To do so, issue the following PROMPT command (PROMPT doesn't care about the case of the letters used):

```
C> PROMPT  $P$G <ENTER>

C:\>
```

The backslash appears between the colon and the greater-than (>) character. This indicates that you are in the root directory of drive C.

The PROMPT command is an internal DOS command that lets you define and customize the DOS system prompt, as you will learn in Chapter 41. Although users sometimes get very creative with their prompts, I recommend using the prompt PG. The characters $P direct DOS to display the current drive and directory, while the characters $G direct DOS to display the greater-than character (>). As you will see, using this system prompt, you can easily determine the current directory.

CHDIR Changes the Current Directory

Changing the current directory is similar to opening a different drawer in a filing cabinet—DOS will look in the open drawer first. The DOS command to change directories is CHDIR. The following, for example, selects the directory DOS as the current directory. The spaces after the command are optional; we will use two spaces for ease of reading and understanding.

```
C:\> CHDIR  \DOS <ENTER>

C:\DOS>
```

Note how the system prompt changes to reflect the current directory. The backslash now has a name after it. If your prompt does not change, type **PROMPT PG** and press ENTER.

The DOS subdirectory is now the current directory. Unless you tell DOS to look elsewhere, DOS will look in this directory by default. For example, if you invoke the DIR command, DIR will display the list of external DOS commands among the files shown here:

```
C:\DOS> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS
```

```
.            <DIR>      11-23-92   9:26p
..           <DIR>      11-23-92   9:26p
DBLSPACE BIN     50284 02-12-93   6:00a
FORMAT   COM     22717 02-12-93   6:00a
NLSFUNC  EXE      7036 02-12-93   6:00a
COUNTRY  SYS     17066 02-12-93   6:00a
  :        :        :      :         :
  :        :        :      :         :
COMMAND  COM     52925 02-12-93   6:00a
QBASIC   INI       132 02-20-93  12:10p
MOUSE    INI        28 02-03-93  11:54a
DEFAULT  CAT        66 02-08-93   7:51p
      129 file(s)    5826105 bytes
                    16756736 bytes free
```

You may recall from before that, to list the files in the DOS directory, you had to specify the directory name DOS in the DIR command, as shown here:

```
C:\> DIR  \DOS <ENTER>
```

The current directory then was the root directory, and you had to tell DOS explicitly to look elsewhere. Now, since you are in the DOS directory, you can simply execute the following command to display the directory entry for FORMAT.COM:

```
C:\DOS> DIR FORMAT.COM <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

FORMAT   COM     22717 02-12-93   6:00a
        1 file(s)      22717 bytes
                    16762880 bytes free
```

DOS locates FORMAT.COM because the file is in the current directory. Thus, you could copy the file to a disk in drive A, as follows:

```
C:\DOS> COPY  FORMAT.COM  A: <ENTER>
         1 File(s) copied
```

As you can see, by changing the current directory, you make it much easier to access files. To change from the DOS directory back to the root, you can use CHDIR as follows:

```
C:\DOS> CHDIR  \ <ENTER>

C:\>
```

Because CHDIR is so frequently used, DOS lets you abbreviate CHDIR as simply CD. Using the CD abbreviation, the following command selects the DOS directory as the current directory.

```
C:\> CD \DOS <ENTER>
```

Now perform a directory listing. Before DIR displays the directory entries, DIR will display a line that tells you the corresponding directory name.

```
C:\DOS> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\DOS

.            <DIR>      11-23-92   9:26p
..           <DIR>      11-23-92   9:26p
DBLSPACE BIN      50284 02-12-93   6:00a
 :         :          :      :         :
MOUSE    INI         28 02-03-93  11:54a
DEFAULT  CAT         66 02-08-93   7:51p
      128 file(s)    5826105 bytes
                    16762880 bytes free
```

If you invoke the CHDIR command without including a directory name, CHDIR will simply display the current directory:

```
C:\DOS> CHDIR <ENTER>
C:\DOS
```

If you include a drive letter, CHDIR (or CD) will display the current directory for the drive specified.

```
C:\DOS> CD  A: <ENTER>
A:\
```

MKDIR Creates Directories

DOS directories exist to help you organize your files. When you install new software packages, most installation programs will create a unique directory for you. In some cases, however, you will need to create your own directories. The MKDIR (make directory) command lets you create directories on your disk. To begin, you can use MKDIR to create a directory named TESTDIR, as shown here:

```
C:\> MKDIR  \TESTDIR <ENTER>
```

Using the DIR command, you can be sure MKDIR created the directory:

```
C:\> DIR  \ <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\

AUTOEXEC BAT       108 03-01-93   9:27p
CONFIG   SYS       102 03-01-93   9:27p
DOS          <DIR>     03-01-93   4:44p
DISKUTIL     <DIR>     02-16-93   4:55p
BATCH        <DIR>     01-28-93  10:13a
TESTDIR      <DIR>     03-03-93  10:47a
        6 file(s)       210 bytes
                   16014400 bytes free
```

In this case, creating the TESTDIR directory in the root is conceptually similar to labeling a filing cabinet drawer "TESTDIR" shown in Figure 10-1.

Using CD (the abbreviation for the CHDIR command), select TESTDIR as the current directory.

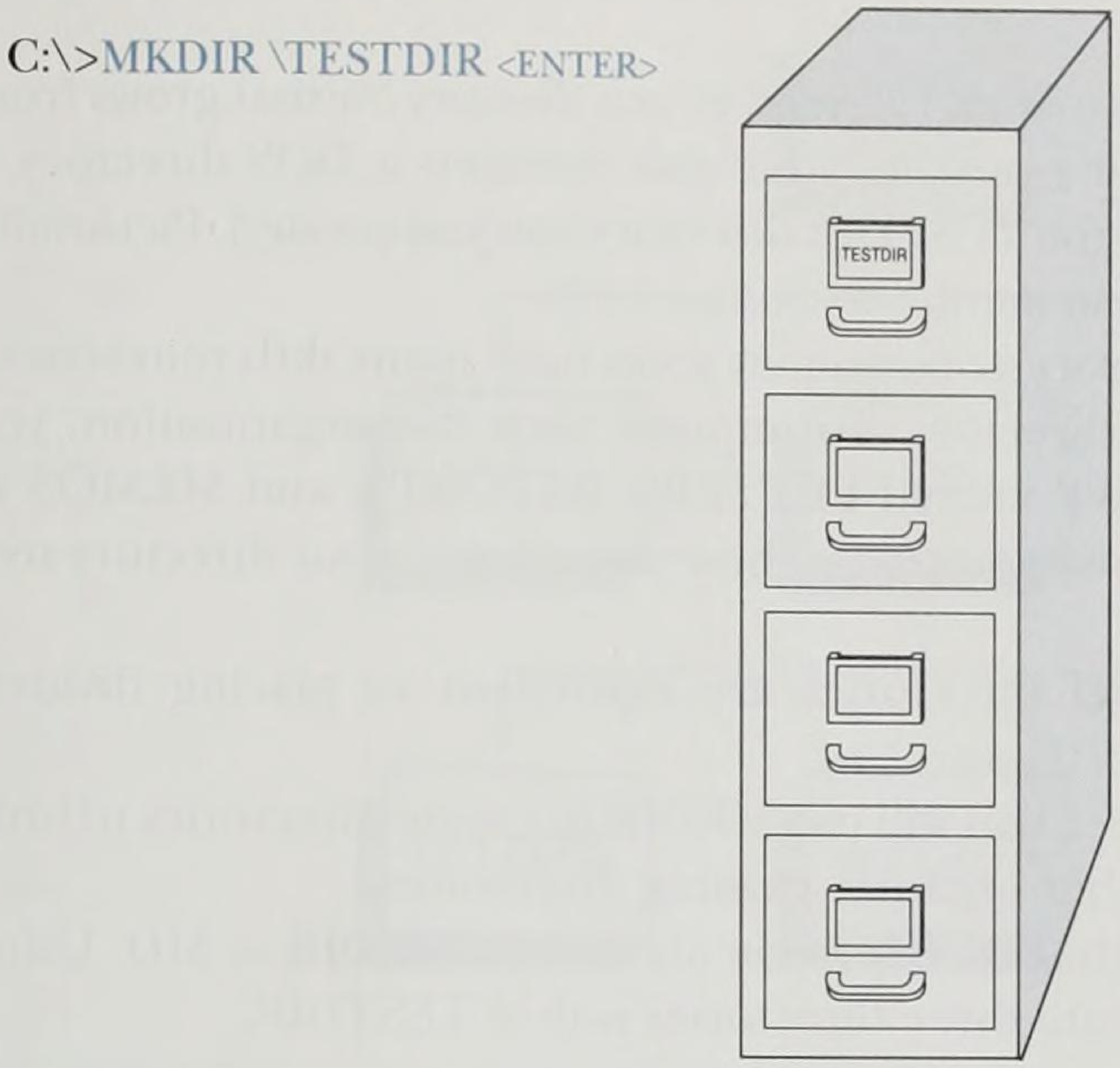


Figure 10-1. *Using MKDIR is similar to adding a new label to a filing cabinet drawer*

```
C:\> CD  \TESTDIR <ENTER>
```

Using DIR, you can display the directory's contents:

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR

.           <DIR>     03-03-93  10:47a
..          <DIR>     03-03-93  10:47a
      2 file(s)           0 bytes
                   16010304 bytes free
```

Every directory you create has two predefined directory entries— a single period (.) and double periods (..) —that advanced DOS users can use to simplify commands.

For now, simply understand that these two directory entries will automatically appear in every directory you create.

The directories on your disk are often referred to as a *directory tree* that grows from the root. Here's why. Assume, for example, your disk contains a DOS directory, a WordPerfect directory (WP), and the TESTDIR directory you just created. Pictorially, your directory structure will appear similar to Figure 10-2.

As you work with your word processor, you may soon have many different letters, reports, and memos in the WP directory. To improve your file organization, you might create directories within WP named LETTERS, REPORTS, and MEMOS to store the corresponding files. When you create these directories, your directory tree grows, as shown in Figure 10-3.

Conceptually, these lower-level directories are equivalent to placing dividers within the WP drawer, as shown in Figure 10-4.

As you work with your computer, you will use MKDIR to create directories to hold new software programs or to further organize existing directories.

MKDIR is so frequently used that DOS lets you abbreviate MKDIR as MD. Using MD, the following commands create three directories within TESTDIR.

```
C:\TESTDIR> MD  \TESTDIR\MEMOS    <ENTER>
C:\TESTDIR> MD  \TESTDIR\LETTERS <ENTER>
C:\TESTDIR> MD  \TESTDIR\REPORTS <ENTER>
```

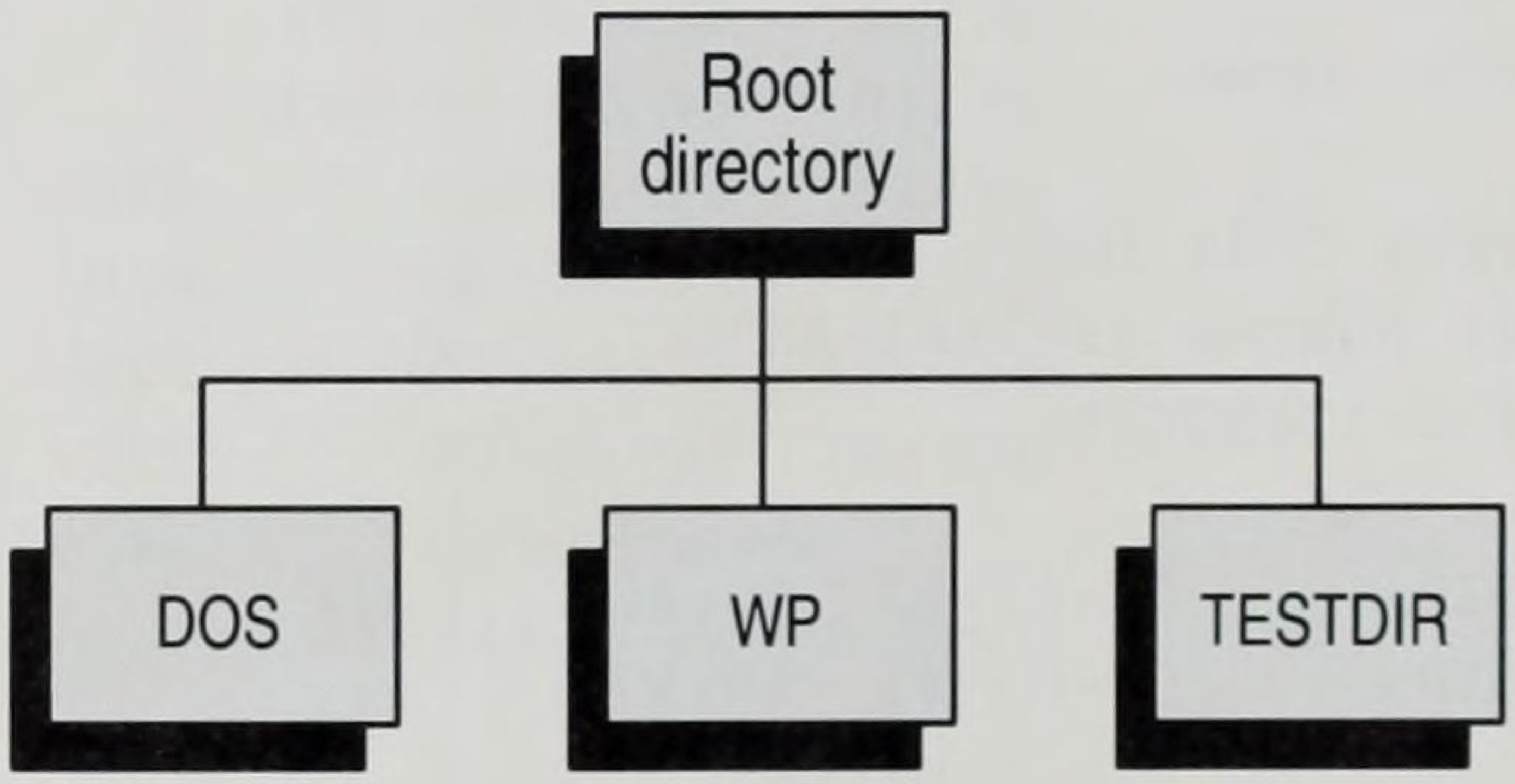


Figure 10-2. *Directory tree for DOS, WP, and TESTDIR*

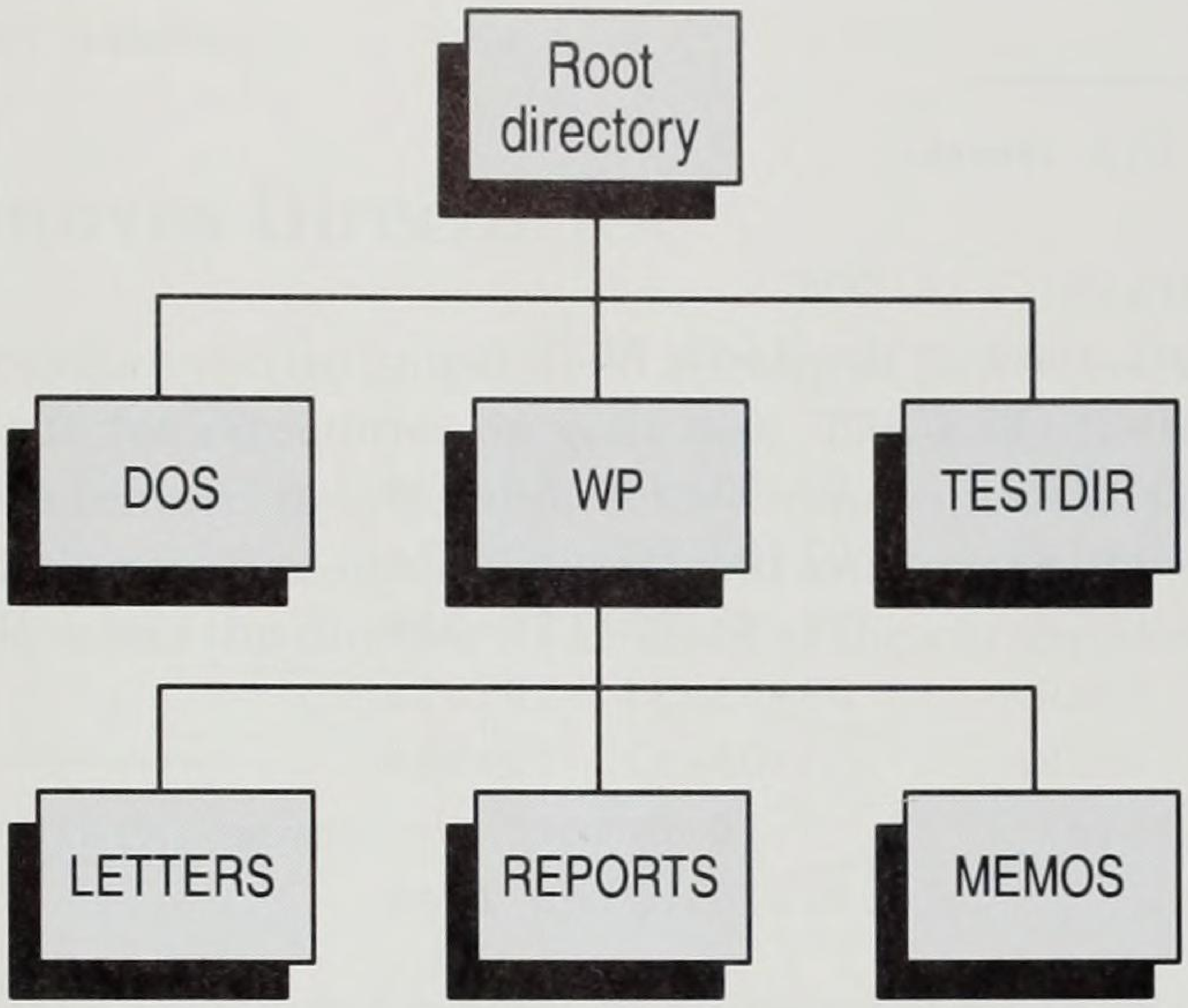

Root directory
DOS
WP
TESTDIR
LETTERS
REPORTS
MEMOS

Figure 10-3. Using additional directories to organize WP

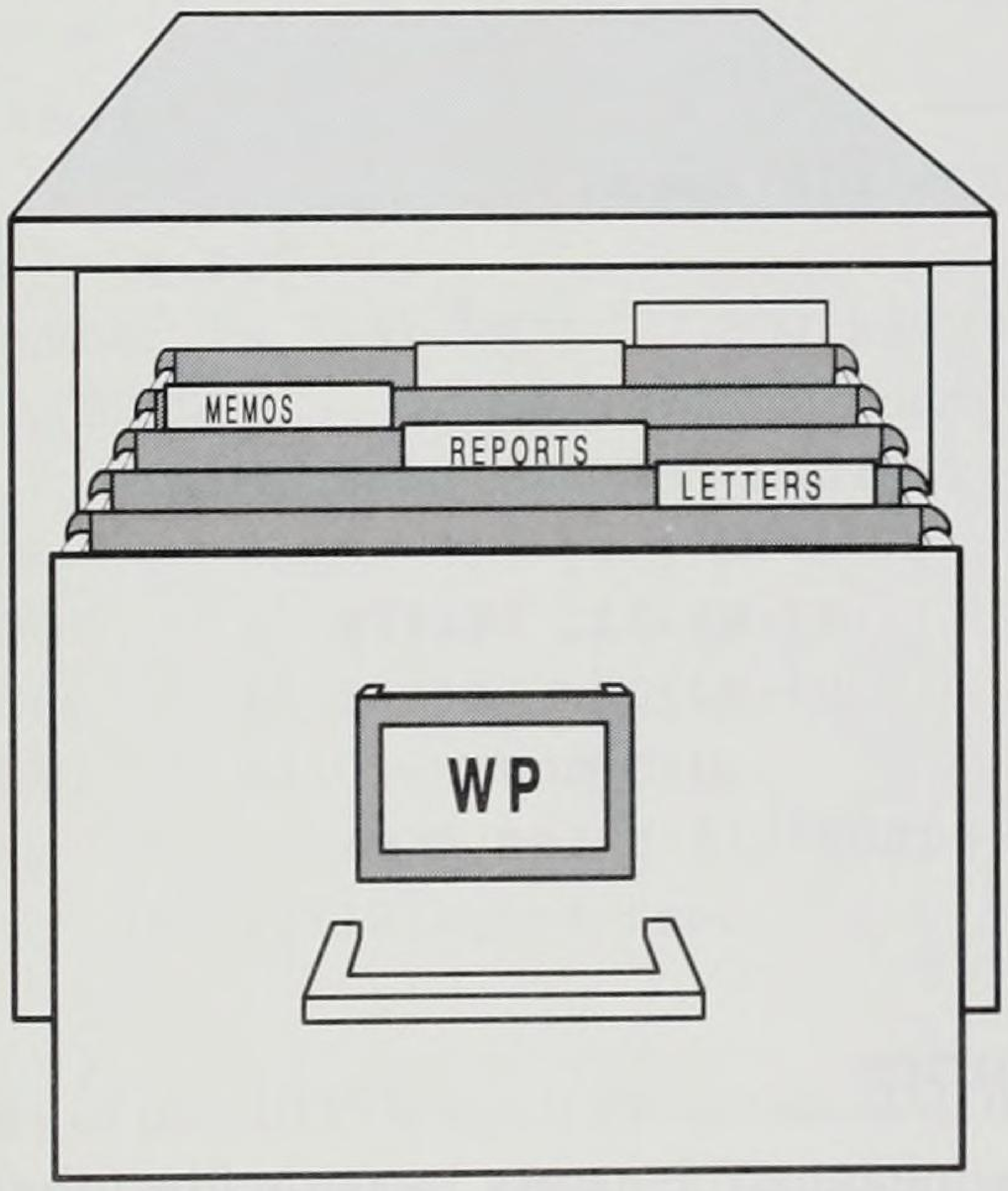

MEMOS
REPORTS
LETTERS
WP

Figure 10-4. Dividing a directory for improved file organization

A directory listing of TESTDIR reveals the three directories:

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR
.            <DIR>     03-03-93  10:47a
..           <DIR>     03-03-93  10:47a
MEMOS        <DIR>     03-03-93  10:51a
LETTERS      <DIR>     03-03-93  10:51a
REPORTS      <DIR>     03-03-93  10:51a
        5 file(s)          0 bytes
                    16098016 bytes free
```

Using CHDIR (CD), you can select the LETTERS directory:

```
C:\TESTDIR> CD  \TESTDIR\LETTERS <ENTER>
```

A directory listing reveals that DOS has, as always, created the directory entries single period (.) and double periods (..) within the directory.

```
C:\TESTDIR\LETTERS> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR\LETTERS

.          <DIR>     03-03-93  10:47a
..         <DIR>     03-03-93  10:47a
      2 file(s)          0 bytes
                  16098016 bytes free
```

Naming Your Directories

As you should assign meaningful names to your files, the same is true for the directories you create. Like DOS filenames, your directories can have an eight-character base name and a three-character extension. In most cases, however, users only use the eight-character base name for directories. As you will recall from Chapter 4, the three-character extension indicates a file's type, which really isn't meaningful with respect to directories. All the characters you can use in DOS filenames are valid for DOS directory names.

RMDIR Removes Directories

As there are times when you no longer need a group of files stored in a filing cabinet, so the same is true for directories on your disk. The RMDIR (remove directory) command lets you remove an unused directory from your disk. The directory must be empty (containing no files or directories) and must *not* be the current directory.

Using CHDIR, select the directory TESTDIR as the current directory.

```
C:\> CD  \TESTDIR <ENTER>
```

Next, the following command removes the directory LETTERS from the directory TESTDIR:

```
C:\TESTDIR> RMDIR  \TESTDIR\LETTERS <ENTER>
```

A directory listing reveals RMDIR has removed the directory:

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR

.            <DIR>      03-03-93  10:47a
..           <DIR>      03-03-93  10:47a
MEMOS        <DIR>      03-03-93  10:51a
REPORTS      <DIR>      03-03-93  10:51a
        4 file(s)          0 bytes
                   167002112 bytes free
```

Due to its frequency of use, DOS lets you abbreviate RMDIR as RD. Using the RD abbreviation, the following command removes the directory MEMOS from the TESTDIR directory:

```
C:\TESTDIR> RD  \TESTDIR\MEMOS <ENTER>
```

RMDIR will not let you remove the current directory or a directory containing files or directory entries. If, for example, you try to remove TESTDIR, RMDIR will display an error message similar to the following:

```
Attempt to remove current directory - \TESTDIR
```

Last, remove the directory REPORTS and then change to the root directory and remove the directory TESTDIR, as shown here:

```
C:\TESTDIR> RD  \TESTDIR\REPORTS <ENTER>
C:\TESTDIR> CD  \ <ENTER>
C:\> RD  \TESTDIR <ENTER>
```

SUMMARY

Root Directory (\)

Every DOS disk begins with one directory called the root, within which you can store files and other directories. The root is so named because, like the branches of a tree, all the directories you later create on your disk to store files grow from the root. DOS uses the backslash character (\) to represent the root.

The CHDIR Command

The current directory contains the list of files and commands that DOS examines by default, unless you tell DOS to look elsewhere. The CHDIR command lets you change the current directory. In fact, CHDIR is an abbreviation for change directory. Conceptually, changing the current directory is the same as closing one drawer of a filing cabinet and opening another. CHDIR is so frequently used that DOS lets you abbreviate CHDIR as CD. The following commands select the DOS directory as the current directory.

```
C:\> CHDIR \DOS <ENTER>

C:\> CD \DOS <ENTER>
```

SUMMARY

If you invoke CHDIR without including a directory name, CHDIR will simply display the current directory.

The MKDIR Command

The MKDIR command lets you create directories on your disk to improve your file organization. In fact, MKDIR stands for make directory. As you work with DOS, you will make directories to hold new software programs and to improve your file organization by further dividing existing directories. Due to its frequency of use, DOS lets you abbreviate MKDIR as MD, The following commands create a directory named BUSINESS.

```
C:\>MKDIR \BUSINESS <ENTER>

C:\>MD \BUSINESS <ENTER>
```

The RMDIR Command

The RMDIR command lets you remove a directory from your disk. In fact, RMDIR stands for remove directory. Before you can remove a directory, it must be empty and cannot be the current directory. Due to its frequency of use, DOS lets you abbreviate RMDIR as RD. The following commands remove a directory named BUSINESS from your disk.

```
C:\>RMDIR \BUSINESS <ENTER>

C:\>RD \BUSINESS <ENTER>
```

Hands On

Using the CHDIR command, select the DOS directory as the current directory.

```
C:\> CHDIR  \DOS <ENTER>
```

Use the following DIR command to display the files, one screenful at a time, in the directory.

```
C:\DOS> DIR  /P <ENTER>
```

Note that DIR displays the directory name at the top of the directory listing.

Next, use the following CD command to select the root directory.

```
C:\DOS> CD  \ <ENTER>
```

Using the DIR command, note the other directories in the root. Use CHDIR to move between each directory and the root.

Next, using MKDIR at the root, create the directory MUSIC, as shown here:

```
C:\> MKDIR \MUSIC <ENTER>
```

Use DIR to verify the directory exists.

```
C:\> DIR <ENTER>
```

Next, using the MD abbreviation, create the directories JAZZ and ROCK within MUSIC, as shown here:

```
C:\> MD   \MUSIC\JAZZ <ENTER>
C:\> MD   \MUSIC\ROCK <ENTER>
```

Use the following DIR command to display the directory entries:

```
C:\> DIR  \MUSIC <ENTER>
```

Next, using RMDIR, remove the directories JAZZ and ROCK.

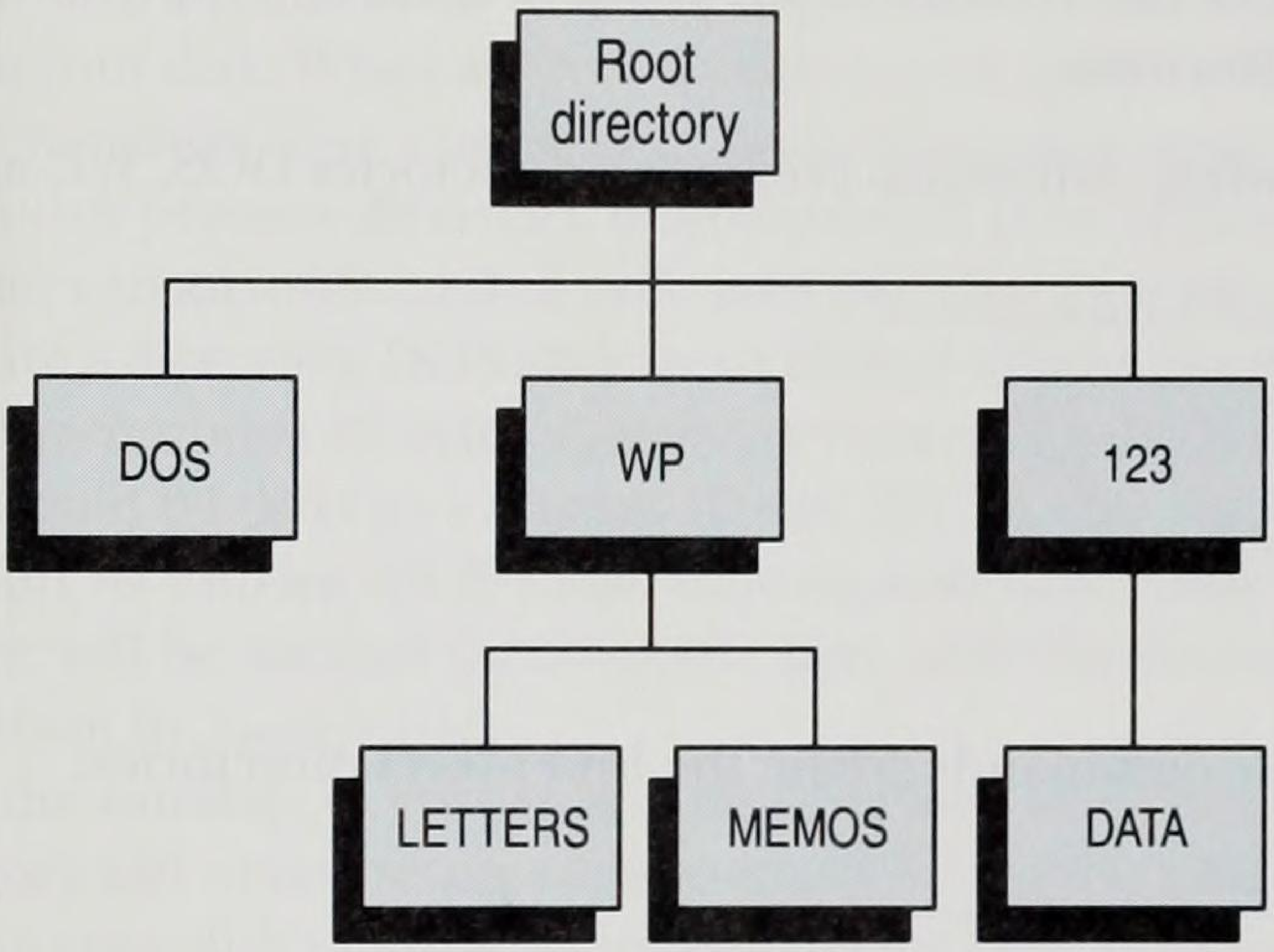
```
C:\> RMDIR  \MUSIC\JAZZ <ENTER>
C:\> RMDIR  \MUSIC\ROCK <ENTER>
```

Last, using the RD abbreviation, remove the MUSIC directory.

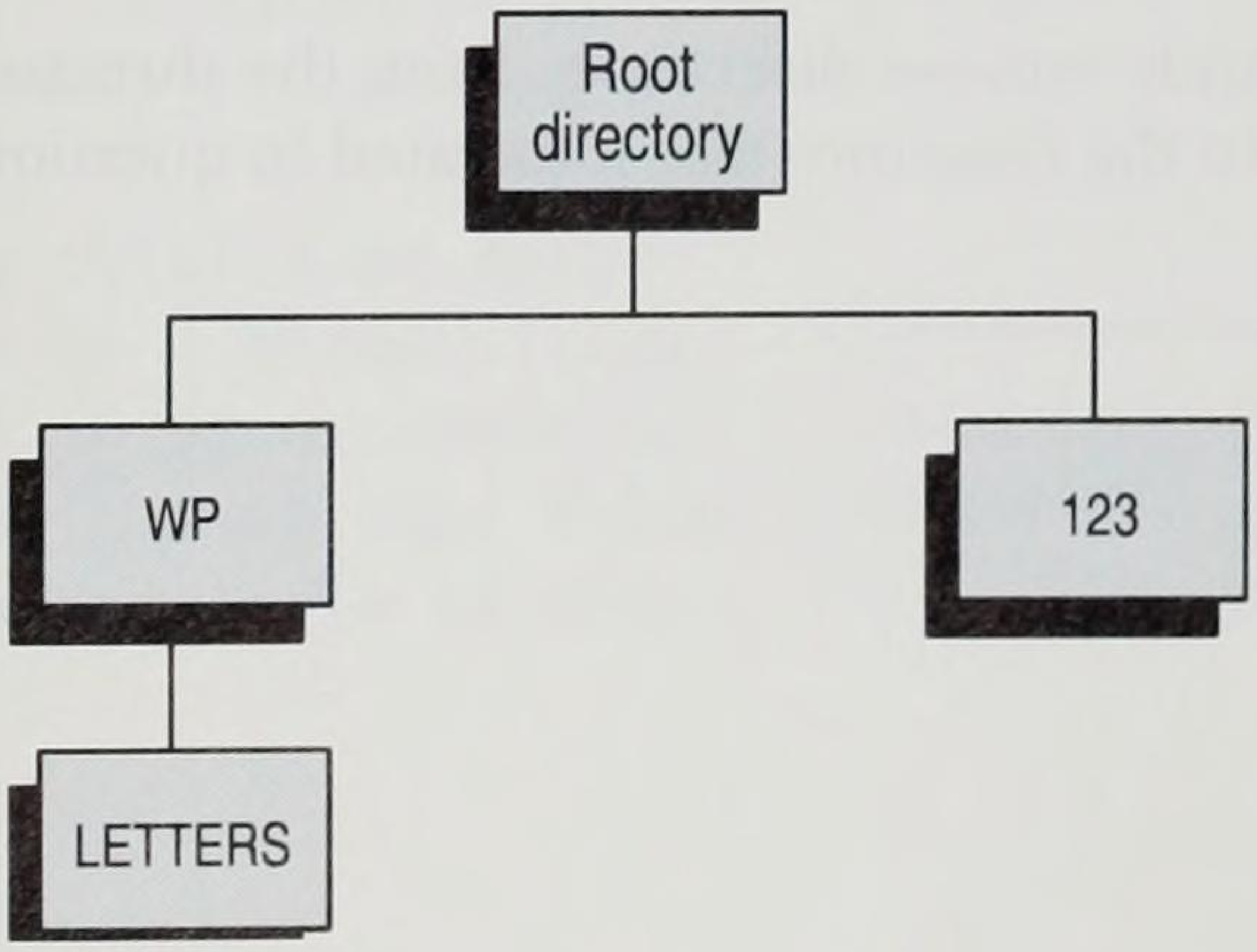
```
C:\> RD  \MUSIC <ENTER>
```

Review

1. What is the root directory?
2. What is the current directory?
3. What are the directory entries single period (.) and double periods (..)?
4. Before DOS can remove a directory, what two conditions must be met?
5. Given a disk containing only the root directory, what commands must you execute to create the directory tree shown here?


Root directory
DOS
WP
123
LETTERS
MEMOS
DATA

6. Given the directory tree shown in question 5, what commands must you execute to produce the directory tree shown here?


Root directory
WP
123
LETTERS

Answers

1. Every DOS disk begins with one directory called the root. The root directory is so named because, like the branches of a tree, all the directories you will later create on your disk grow from the root. DOS uses the backslash character (\) to represent the root.
2. The current directory is the list of files and commands that, by default, DOS searches unless you tell DOS to look elsewhere. The current directory is conceptually the same as the open filing cabinet drawer.
3. Every directory you create on your disk will have the . and .. entries. You can use these as directory name abbreviations as you execute advanced DOS commands. For now, simply know that every directory you create contains these two entries.
4. Before DOS can remove a directory, it must be empty and must not be the current directory.
5. The following commands create the directories DOS, WP, and 123:

```
A:\> MKDIR \DOS
A:\> MKDIR \WP
A:\> MKDIR \123
```

   Next, these commands create the lower-level directories:

```
A:\> MD \WP\LETTERS
A:\> MD \WP\MEMOS
A:\> MD \123\DATA
```

6. These commands remove directories from the directory tree in question 5, which results in the directory tree illustrated in question 6:

```
A:\> RMDIR \123\DATA
A:\> RMDIR \WP\MEMOS
A:\> RMDIR \DOS
```

Advanced Concepts

Although most users know they should use directories to improve their file organization, many users ask how many files each directory should contain. With the exception of the root directory, DOS directories can hold an almost limitless number of files. As the number of files in a directory increases, so will your difficulty in locating a specific file. In addition, the more files a directory contains, the more time DOS will need to perform directory search operations. In this way, long directory listings decrease your system performance.

If you are installing a large software program, you may not have any control over the number of files in the directory. If you are creating directories for your own files, however, you can control the number of files the directory contains. Chapter 42 covers the subject of disk fragmentation, which occurs when a file's contents are stored in locations spread across your disk. Fragmented files decrease your system performance because DOS must wait for each sector of the disk containing the file to rotate past the disk's read/write head. It is possible for directories to become fragmented across your disk. When a directory becomes fragmented, every operation on that directory becomes time consuming, decreasing your system performance. The only way you can prevent directory fragmentation is to restrict the number of entries in a directory to an amount that only uses one disk allocation unit or cluster.

When you create a directory, DOS allocates a cluster to hold the directory entries. Each directory entry requires 32 bytes. Assuming your disk uses 2048-byte clusters, a single cluster can hold 64 directory entries (2048/32). In this case, if the directory contains more than 64 entries, DOS must allocate a second cluster. In most cases, this second cluster will be located far from the first, and the directory will become fragmented, as shown in Figure 10-5.

If you restrict the number of entries in your directory to an amount that fits in a cluster, the directory will never become fragmented. Using the CHKDSK command, you can determine your disk's cluster (allocation unit) size, as shown here:

```
C:\> CHKDSK <ENTER>

Volume DOS          created 04-16-1993 4:43p
Volume Serial Number is 1661-AB75

 200065024 bytes total disk space
     77824 bytes in 2 hidden files
    397312 bytes in 93 directories
  32583680 bytes in 1735 user files
 167002112 bytes available on disk
```

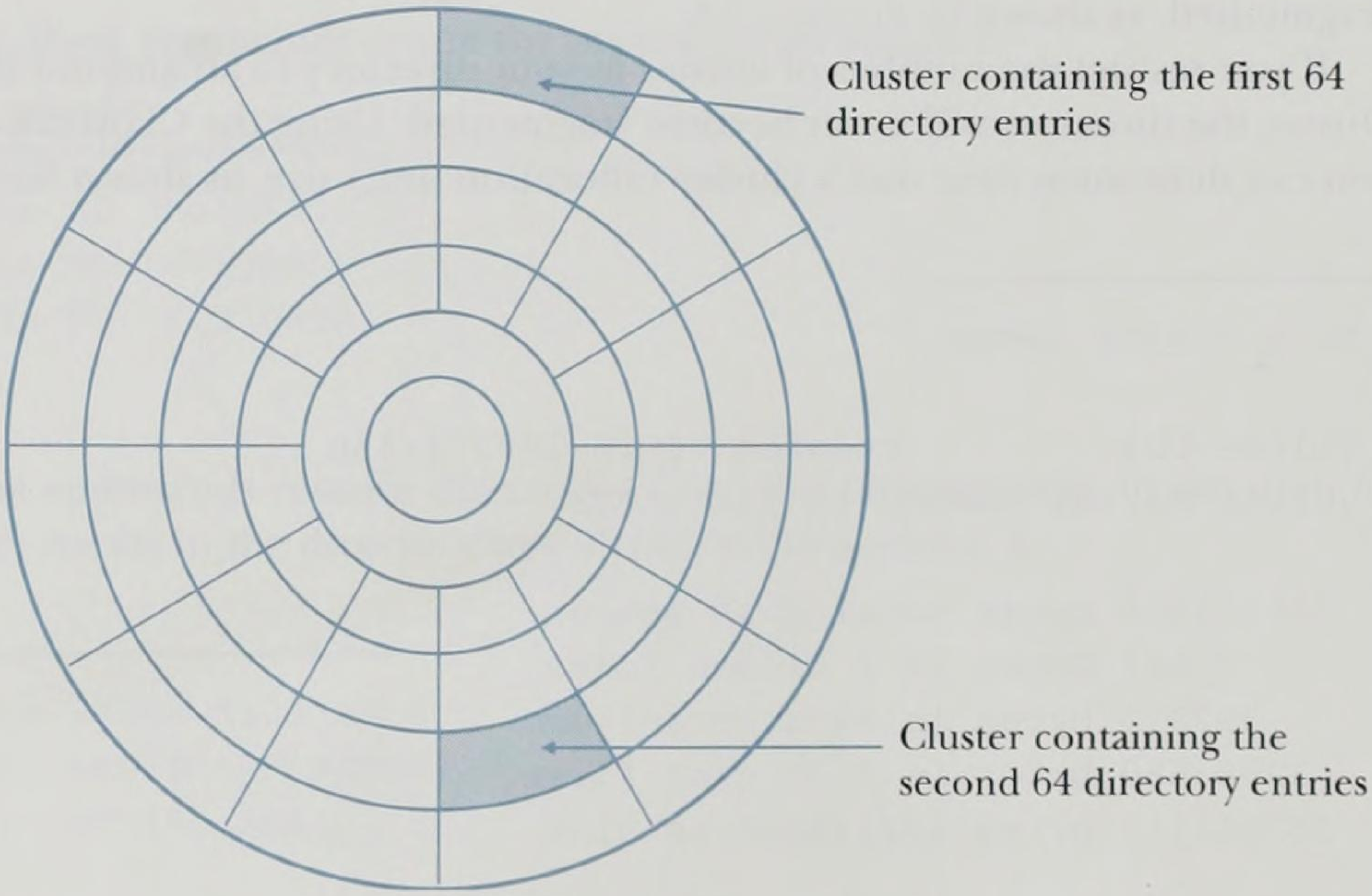
```
      4096 bytes in each allocation unit
     48844 total allocation units on disk
     40772 available allocation units on disk

    655360 total bytes memory
    578560 bytes free
```

Next, divide this amount by 32 (in this example, 4096/32) to determine the number of directory entries a cluster can store. Use this value as a rule of thumb for determining the number of entries your directory should contain.

Protecting Private Directories

If you work in an office where your PC is accessible to other users, you may want to use an extended ASCII character to protect your directory from access by other users. In addition to the letters A through Z and the numbers 0 to 9, DOS lets you use extended ASCII characters in your file and directory names. Appendix B presents the entire ASCII and extended ASCII character sets.

In this case, you can use the extended ASCII value 255 to create a special blank character at the end of your directory name. To create this character, you must hold down the ALT key and keep it down while typing **255** with your keyboard's *numeric*


Cluster containing the first 64 directory entries

Cluster containing the second 64 directory entries

Figure 10-5. Creating a fragmented directory

keypad. You must use the keypad—it won't work with the number keys in the typewriter section.

For example, you can create a directory named PRIVATE that has the special blank character appended. To do so, type the following command but don't press ENTER.

```
C:\> MKDIR PRIVATE_
```

Next, hold down the ALT key, keep it down, and type 255. You must use the numeric keypad. When you release the ALT key, you will see the cursor move one position to the right indicating the blank character's presence.

```
C:\> MKDIR PRIVATE _
```

Press ENTER to create the directory. If another user examines your disk, the directory name will appear in a directory listing.

```
C:\> DIR  \ <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\

AUTOEXEC BAT       108 03-01-93   9:27p
CONFIG   SYS       102 03-01-93   9:27p
DOS          <DIR>     02-16-93   4:44p
DISKUTIL     <DIR>     03-16-93   4:55p
BATCH        <DIR>     01-28-93  10:13a
PRIVATE      <DIR>     03-03-93  10:47a
        6 file(s)       210 bytes
                   16014400 bytes free
```

However, if the user tries to access the directory without including the special blank character, the command will fail. This even holds true for wildcard searches. PRIVATE*, P*, and PRIVATE? will all fail. Some menu or graphical interfaces, however, might allow access to the directory.

```
C:\> CHDIR  \PRIVATE <ENTER>
Invalid directory
```

When you need to access the files in the directory, you simply use ALT-255 to append the special blank to the name as required.

Every Drive Has a Current Directory

If you ask users "How many current directories are there?," most will say one. Actually, there is a current directory for each disk drive. When DOS starts, the current directory for each drive is initially the drive's root directory. If you place a disk in drive A and use CHDIR to display the disk's current directory, CHDIR will display the root.

```
C:\> CHDIR  A: <ENTER>
A:\
```

If you then change the current directory on drive C to DOS, drive A's current directory remains the root.

```
C:\> CHDIR  \DOS <ENTER>
C:\DOS> CHDIR  A: <ENTER>
A:\
```

As you can see, DOS keeps track of each disk's current directory. If you change the disks in a floppy drive, DOS automatically selects the root directory as the current directory if the new disk does not have a directory corresponding to the previous directory name.

Moving Files From One Directory or Disk to Another

Prior to DOS 6, the only way to move a file from one disk or directory to another was to use the COPY and DEL commands to first duplicate and then erase the file. Beginning with version 6, however, DOS provides the MOVE command that lets you quickly perform such operations. The format of the MOVE command is

```
MOVE [drive:][path]SourceFile [drive:][path]TargetFile
```

The *SourceFile* specifies the file you want to move. The *TargetFile* specifies the desired file location. The following command, for example, moves the file MYBACKUP.BAT from the current directory into the BATCH directory.

```
C:\SUBDIR> MOVE MYBACKUP.BAT \BATCH\*.* <ENTER>

c:\subdir\mybackup.bat => c:\batch\mybackup.bat [ok]
```

The MOVE command fully supports DOS wildcards. The following command, for example, moves all of the current directory batch files into the BATCH directory.

```
C:\SUBDIR> MOVE *.BAT \BATCH\*.* <ENTER>
```

You can also use MOVE to switch a file from one disk to another. The following command moves the file FILENAME.EXT from the current directory to the disk in drive B.

```
C:\SUBDIR> MOVE FILENAME.EXT B:*.* <ENTER>
```

Finally, you can use the MOVE command to rename a directory! For example the following command renames the directory OLD_DIR to NEW_DIR.

```
C:\> MOVE \OLD_DIR \NEW_DIR <ENTER>
```

Key Terms

Directory tree The directory organization of your disk, beginning at the root directory and growing outward, much like the branches of a tree.

Root directory The first level in the directory tree, from which all the rest of your directories grow.

Current directory The list of files and commands that DOS searches by default, unless you tell DOS to look elsewhere.

Chapter 11

Getting the Most from DOS Directories

In Chapter 10 you learned that DOS directories help you organize the files you store on your disk. Using the CHDIR, MKDIR, and RMDIR commands, you were able to build and traverse directories. As you learned, your directories grow from an original directory called the root. In this chapter you will learn how to use the TREE command to display your disk's directory structure and to optionally list the files each directory contains, as well as the PATH command to simplify the execution of your commonly used commands. In addition, if you are using DOS 3.3 or later, you will learn how the FASTOPEN command can improve system performance. Finally, this chapter takes a look at the single period (.) and the double periods (..) directory abbreviations.

Displaying Your Directory Tree

The directories you create on your disk grow from the root directory much like the branches of a tree, as shown in Figure 11-1.

As you can imagine, after you work with DOS for some time, your directory structure can become very large and complicated. To help you track the directories on your disk, DOS provides the TREE command. TREE displays a "tree-like" representation of your directory structure. TREE is an external DOS command in the DOS directory. Use CHDIR to select the DOS directory and then issue the TREE command.

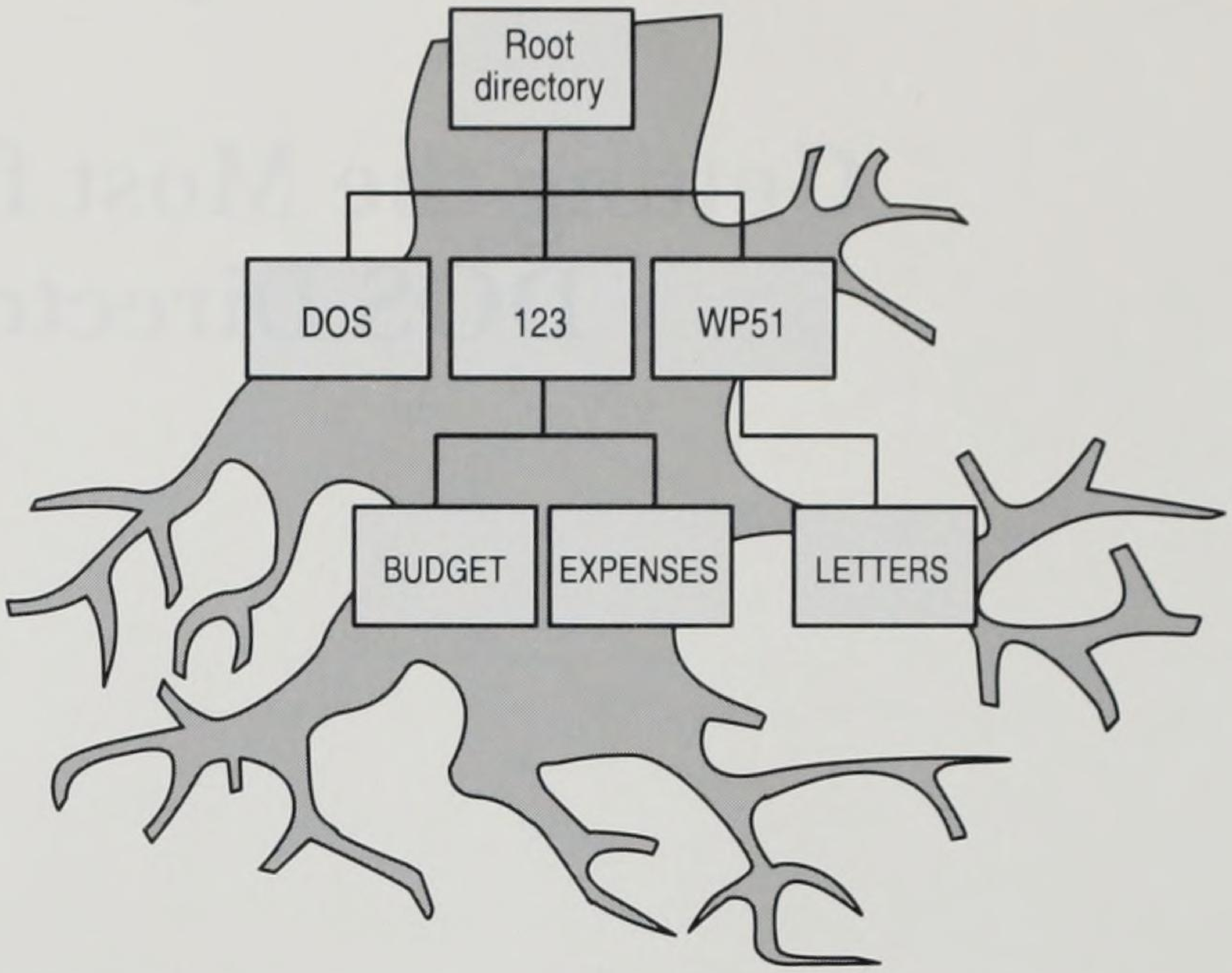


Figure 11-1. *Your disk's directory structure forms your disk's directory tree*

```
C:\> CHDIR  \DOS <ENTER>
C:\DOS> TREE  C:\ <ENTER>
```

DOS will display your disk's directories on screen. Assume, for example, that the disk in drive A contains the directory structure shown in Figure 11-2.

Under DOS versions prior to 4, the output produced by the TREE command will be similar to the following:

```
C:\> TREE  A: <ENTER>

DIRECTORY PATH LISTING
Path: \MEMOS
Sub-directories:  None

Path: \REPORTS
Sub-directories:  BUDGET
                  EXPENSES

Path: \REPORTS\BUDGET
```

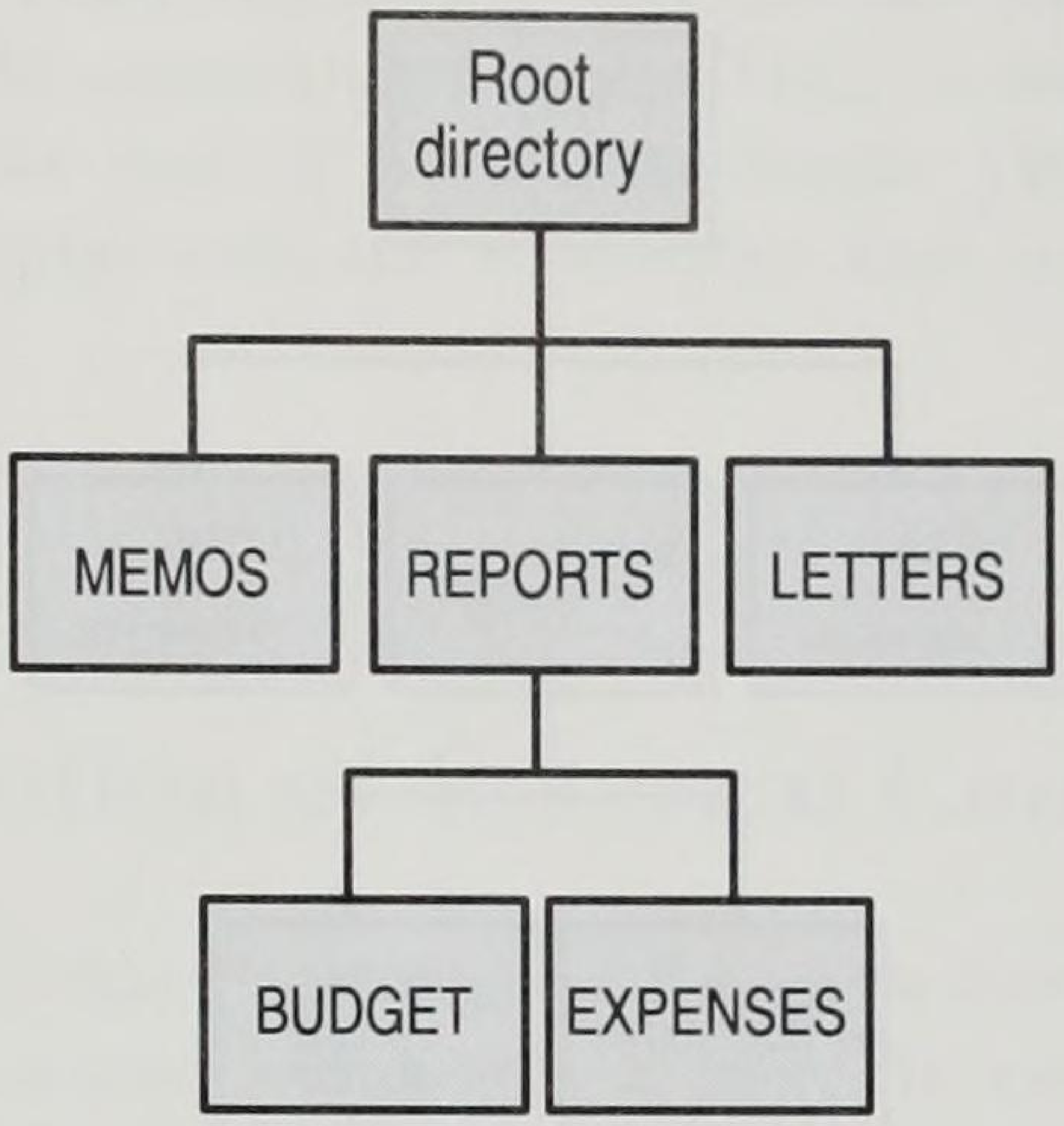


Figure 11-2. *A simple directory structure*

```
Sub-directories:  None

Path: \REPORTS\EXPENSES
Sub-directories:  None

Path: \LETTERS
Sub-directories:  None
```

Beginning with DOS 4, the TREE command uses a more pictorial representation, shown here:

```
C:\> TREE A: <ENTER>
Directory PATH listing
Volume Serial Number is 0315-1DC9
A:\
├──MEMOS
├──REPORTS
│  ├──BUDGET
│  └──EXPENSES
└──LETTERS
```

By default, the TREE command only displays directory names. If you include the /F switch, TREE will display all the files in each directory. Assume, for example, the directories on the disk in drive A contain the files shown in Figure 11-3.

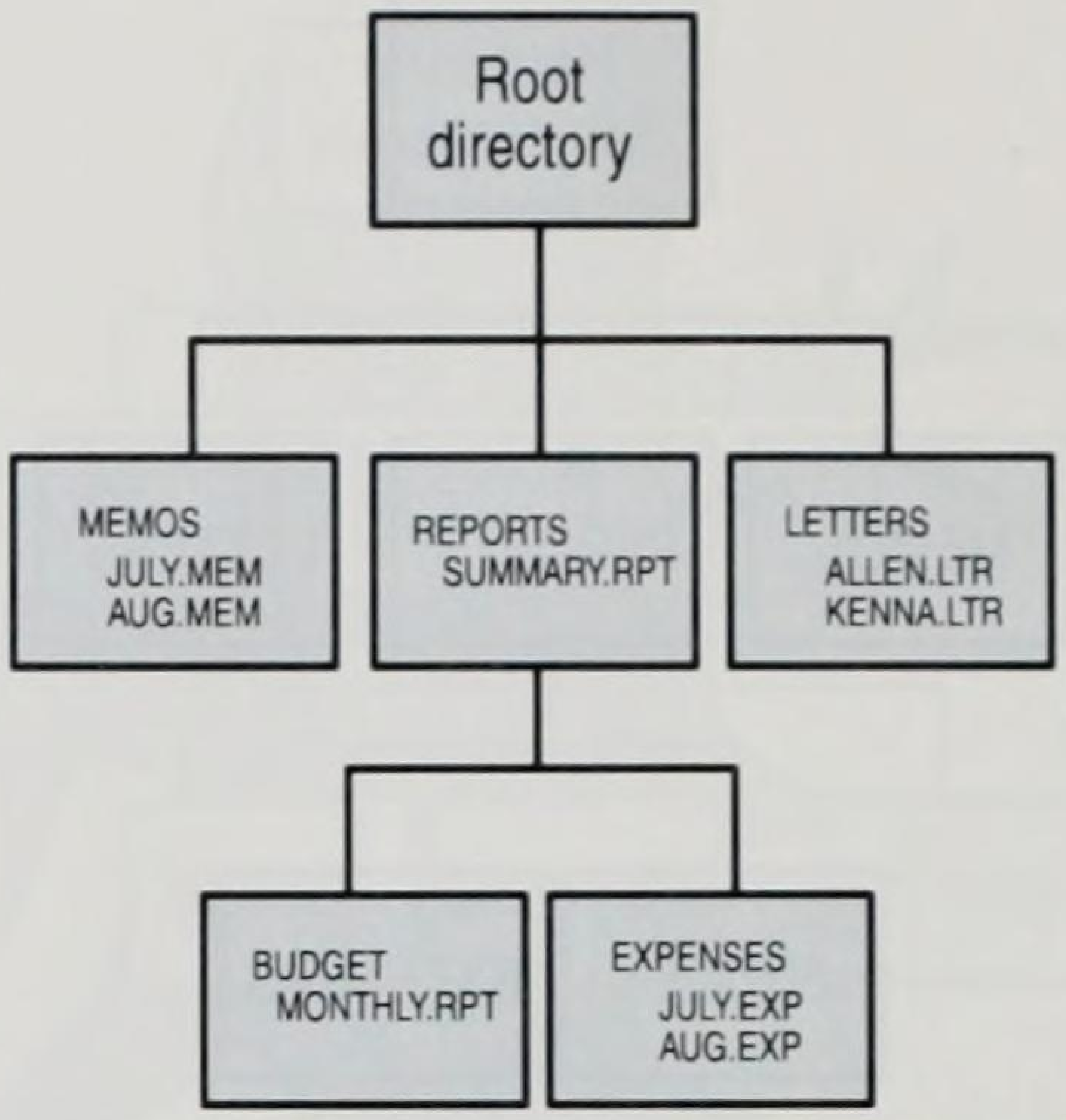

Root directory
MEMOS
JULY.MEM
AUG.MEM
REPORTS
SUMMARY.RPT
LETTERS
ALLEN.LTR
KENNA.LTR
BUDGET
MONTHLY.RPT
EXPENSES
JULY.EXP
AUG.EXP

Figure 11-3. *Directory tree with files*

In DOS 4 or later, using the /F switch with TREE results in the following:

```
C:\> TREE A: <ENTER>
Directory PATH listing
Volume Serial Number is 0315-1DC9
A:\
├───MEMOS
│       JULY.MEM
│       AUG.MEM
│
├───REPORTS
│   │
│   │   SUMMARY.RPT
│   │
│   ├───BUDGET
│   │       MONTHLY.RPT
│   │
│   └───EXPENSES
│           JULY.EXP
│           AUG.EXP
│
└───LETTERS
        ALLEN.LTR
        KENNA.LTR
```

Each of the previous TREE commands specified the root directory in the command line. In so doing, the commands directed TREE to display the disk's directory structure beginning at the root. If you simply invoke TREE without specifying a directory, TREE will display only the directories that reside below the current directory.

Simplifying Execution of External Commands

You learned in Chapter 10 that when you install a new software package on your hard disk, you typically create a unique directory to store the program. In most cases, to use the program, you will use CHDIR to select the directory containing the program's files. For example, to use the WordPerfect word processor, you would select the WP51 directory and invoke WordPerfect 5.1 as shown here:

```
C:\> CHDIR  \WP51 <ENTER>
C:\WP51> WP <ENTER>
```

In this case, by selecting the WP51 directory before you begin, you keep your word processing files in one location.

In some cases, however, a directory may contain several commands you use so often that you don't want to select the directory each time you need to execute one of the commands. The DOS directory, which contains your external commands, is a good example of such a directory. Because of their frequent use, you don't want to have to select the DOS directory as the current directory each time you execute a DOS command.

As you have learned, to access files that are in a DOS directory, you have two choices. First, you can use CHDIR to select the file's directory and then reference the file by name. Second, you can access the file from any directory by specifying a complete directory path name to the file. The same two options apply to executing external DOS commands. If, for example, you want to execute the TREE command to display the directory structure, select the DOS directory and invoke TREE C:\ as in the next command, as shown here:

```
C:\> CHDIR  \DOS <ENTER>
C:\DOS> TREE  C:\ <ENTER>
```

Or, you can specify a complete path name to the TREE command, invoking TREE from any directory, as shown here:

```
C:\> C:\DOS\TREE  C:\ <ENTER>
```

By specifying a complete path name to your external commands, you can execute your COM and EXE programs from any directory.

To execute your applications, such as your word processor or spreadsheet, you will normally use CHDIR to select the directory containing the program's files. To execute your commonly used DOS commands, however, repeatedly changing to the DOS directory could become quite time consuming. Fortunately, DOS provides the PATH command, which lets you tell DOS where it should look to find your commonly used external commands.

The PATH Command

Although your disk may contain many different directories, it's very likely only a few of the directories contain commands you use on a regular basis. If you don't tell DOS explicitly, by full path name, to look elsewhere, DOS only searches the current directory for external commands. If your file is in another directory, it is not found. To make it easier for you to execute your commonly used external commands, the PATH command lets you specify a list of directories you want DOS to automatically search each time it fails to locate an external command in the current directory. When issuing an external command in another directory that has been included in a PATH command, you don't have to supply a directory name or change to that directory—DOS will find the command and execute it. Most users, at a minimum, tell PATH to look in the DOS directory for DOS external commands. Issue the following PATH command to determine whether or not you have a command path defined:

```
C:\> PATH <ENTER>
```

For most systems, DOS will display one or more directory names, as shown here:

```
C:\> PATH <ENTER>
PATH=C:\DOS;C:\BATCH
```

If no command path is defined, PATH will display a message stating so:

```
C:\> PATH <ENTER>
No path
```

If your system doesn't have a command path defined, issue the following PATH command directing DOS to automatically search the DOS directory for your external commands.

```
C:\DOS> PATH  C:\DOS <ENTER>
```

Next, select the root directory as the default.

```
C:\DOS> CHDIR  \ <ENTER>
```

Next, with the root directory as the default, issue the following TREE command:

```
C:\> TREE  C:\ <ENTER>
```

Although the external TREE command is in a different directory, DOS finds it and displays your disk's directory structure as desired. Here's why. Each time you execute a command, DOS first checks if the command specified is an internal command, such as CLS or DATE, that always resides in your computer's fast electronic memory. If the command is internal, DOS immediately executes it; otherwise, DOS searches the files in the current directory for a matching EXE or COM file. If DOS finds a matching external command, it executes it; otherwise, DOS checks to see if you have used PATH to define a command path. As you just read, the command path specifies one or more directories you want DOS to automatically search for a matching external command. In this case, using the command path, DOS locates the TREE command in the DOS directory. As a result, DOS lets you execute TREE without regard for the current directory. The previous example assumed C:\DOS was the only directory in the command path. If your path contains several directories, DOS separates the directory names using semicolons, as shown next.

```
C:\> PATH <ENTER>
PATH=C:\DOS;C:\BATCH
```

In this case, if DOS fails to locate an external command in the current directory, DOS will first search the directory DOS on drive C. If DOS finds a matching command, DOS will execute it; otherwise, DOS will search directory BATCH. If DOS finds a matching command anywhere along the way, DOS executes the command, and the search is done. If DOS fails to locate a matching command in any of the directories, DOS will display an error message indicating this:

```
Bad command or file name
```

Figure 11-4 illustrates the steps DOS performs when you execute a command, starting with its search of internal commands, followed by commands in the current directory, and ending with its traversal of directories listed in the command path.

Changing the Command Path

PATH lets you change, display, and even remove the command path. In Chapter 18, you will learn about a special file named AUTOEXEC.BAT that you can use to specify commands you want DOS to automatically execute each time your system starts. You may already have an AUTOEXEC.BAT file in your root directory. Using the TYPE command, you can display the file's contents.

```
C:\> TYPE  \AUTOEXEC.BAT <ENTER>
```

Most users determine two or three directories that contain their most commonly used commands and include those directory names in the command path. For example, the following PATH command directs DOS to look in the DOS, BATCH, and DISKUTIL directories on drive C for commands it can't find in the current directory:

```
C:\> PATH  C:\DOS;C:\BATCH;C:\DISKUTIL <ENTER>
```

There are no spaces between directory names. Use the semicolon as shown to separate directory names.

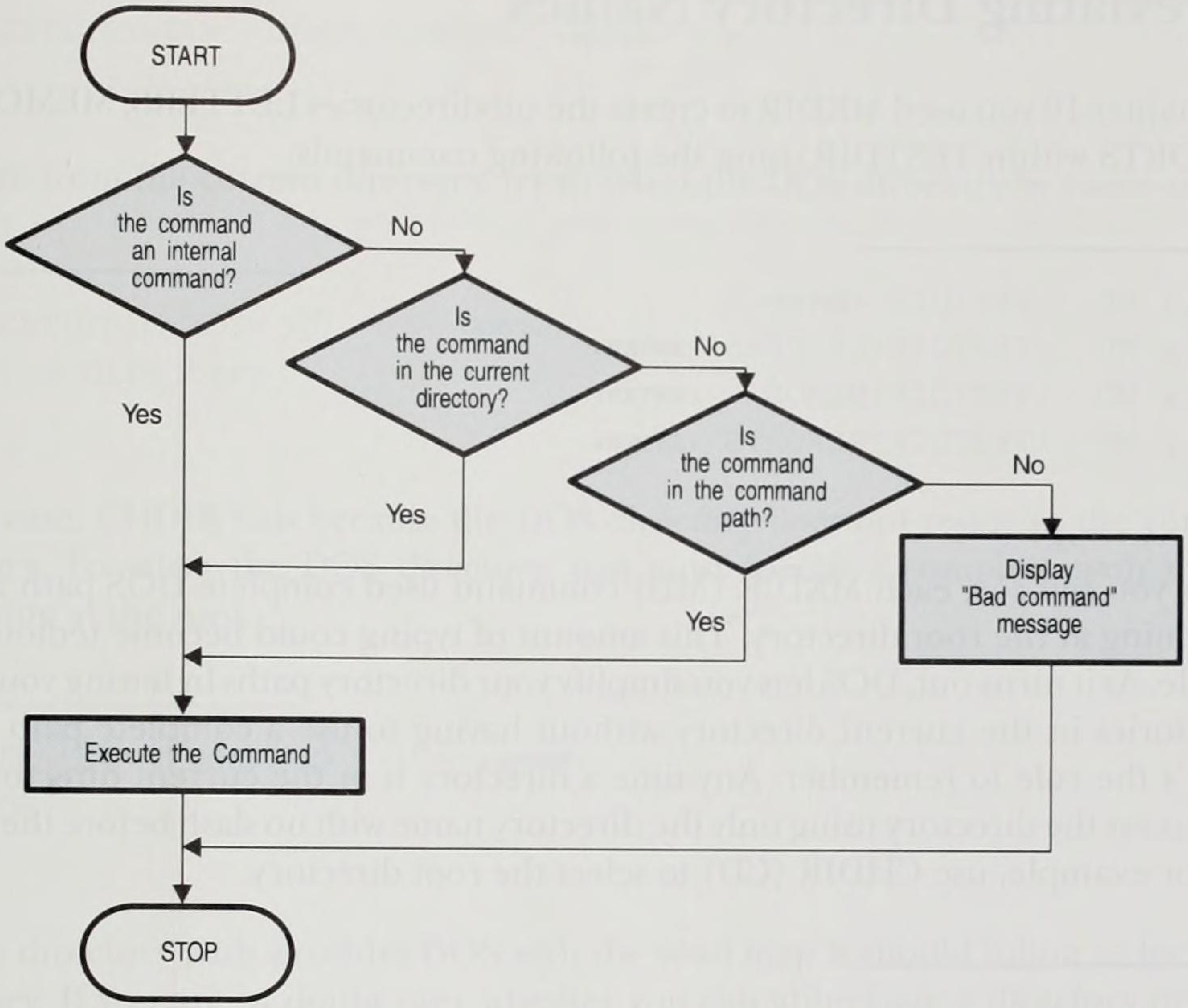

START
Is the command an internal command?
No
Is the command in the current directory?
No
Yes
Is the command in the command path?
No
Yes
Display "Bad command" message
Yes
Execute the Command
STOP

Figure 11-4. *The steps DOS follows to execute your commands*

Note that each entry specifies drive C before the directory name. If you don't include the drive letter, DOS will try to find the directories specified on the current drive. As long as drive C is the current drive, DOS will successfully locate the directories. However, if you change drives, DOS will look on the wrong drive for the directories and will be unable to locate the commands without drive specifiers in the PATH statement. Include drive letters in the command path.

By placing your PATH command in the AUTOEXEC.BAT batch file, you don't have to worry about setting your command path yourself. DOS will automatically set up your path for you each time your system starts. Chapter 18 discusses using PATH within AUTOEXEC.BAT.

Many software programs you buy direct you to include their directory in the command path. In most cases, the installation program you run to put the program on your hard disk will automatically update the PATH command in your AUTOEXEC.BAT file for you. It's a good idea to invoke PATH after you install a new program to determine if your command path has changed. You only want to include in the command path directories that contain commonly used commands. If your command path includes too many directories, your system will perform more slowly because DOS will spend a lot of time simply searching through directories.

Abbreviating Directory Names

In Chapter 10 you used MKDIR to create the subdirectories LETTERS, MEMOS, and REPORTS within TESTDIR using the following commands.

```
C:\> MD  \TESTDIR <ENTER>
C:\> MD  \TESTDIR\LETTERS <ENTER>
C:\> MD  \TESTDIR\MEMOS   <ENTER>
C:\> MD  \TESTDIR\REPORTS <ENTER>
```

As you can see, each MKDIR (MD) command used complete DOS path names, beginning at the root directory. This amount of typing could become tedious after a while. As it turns out, DOS lets you simplify your directory paths by letting you access directories in the current directory without having to use a complete path name. Here's the rule to remember: Any time a directory is in the current directory, you can access the directory using only the directory name with no slash before the name.

For example, use CHDIR (CD) to select the root directory.

```
C:\DOS> CD \ <ENTER>
```

Assuming the TESTDIR directory exists, you can select TESTDIR using CHDIR as follows:

```
C:\> CD  TESTDIR <ENTER>
```

Note that the command does not include a slash before TESTDIR's name. Because the current directory was the root, and because TESTDIR resides immediately below the root, you can access the directory by name without the slash.

With TESTDIR as the current directory, issue the following CHDIR command to select the directory MEMOS:

```
C:\TESTDIR> CD  MEMOS <ENTER>
```

Again, because the MEMOS directory is in the current directory, you can access the directory by name. You could have used the complete directory path name as shown here; however, doing so requires more typing.

```
C:\TESTDIR> CD \TESTDIR\MEMOS <ENTER>
```

Next, from the current directory, try to select the DOS directory by name only.

```
C:\TESTDIR\MEMOS> CD DOS <ENTER>
Invalid directory
```

In this case, CHDIR fails because the DOS directory does not reside in the current directory. To select the DOS directory, you must specify a complete path name, beginning at the root.

```
C:\TESTDIR\MEMOS> CD \DOS <ENTER>
C:\DOS>
```

The directory path provides DOS with the road map it should follow to locate a directory. If you are in doubt over whether you can abbreviate a directory or not, keep in mind that specifying a complete path name will always be correct.

The . and .. Abbreviations

You learned in Chapter 10 that every directory you create automatically has the two directory entries single period (.) and double periods (..), as shown here:

```
C:\TESTDIR> DIR <ENTER>
 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR

.            <DIR>     03-17-93   10:22p
..           <DIR>     03-17-93   10:22p
LETTERS      <DIR>     03-17-93   10:23p
MEMOS        <DIR>     03-17-93   10:23p
REPORTS      <DIR>     03-17-93   10:23p
        5 file(s)             0 bytes
                       16761024 bytes free
```

These two directory entries actually serve as abbreviations for the current directory name and the name of the directory immediately above the current directory, which is the current directory's *parent*. Using CHDIR, select TESTDIR as the current directory. The single period (.) now corresponds to the current directory TESTDIR, while the double periods (..) correspond to TESTDIR's parent, the root. You can verify this with DIR. First, list the files in the current directory using the period:

```
C:\TESTDIR> DIR  . <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\TESTDIR

.            <DIR>       03-17-93     10:22p
..           <DIR>       03-17-93     10:22p
LETTERS      <DIR>       03-17-93     10:23p
MEMOS        <DIR>       03-17-93     10:23p
REPORTS      <DIR>       03-17-93     10:23p
        5 file(s)              0 bytes
                        16761024 bytes free
```

Next, list the files in the parent using the double periods (..):

```
C:\TESTDIR> DIR  .. <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\

DOS          <DIR>       02-16-93   4:44p
DISKUTIL     <DIR>       03-16-93   4:55p
WS           <DIR>       02-18-93   9:25a
BATCH        <DIR>       01-28-93  10:13a
AUTOEXEC BAT         128 03-04-93   1:37p
CONFIG   SYS         128 03-04-93   2:44p
TESTDIR      <DIR>       03-17-93  10:22p
        7 file(s)           256 bytes
                       16756928 bytes free
```

Using CHDIR, select the directory MEMOS.

```
C:\TESTDIR> CD  MEMOS <ENTER>
```

When you change directories, DOS will update the abbreviations. In this case, the single-period abbreviation now corresponds to the current directory MEMOS, while the double periods correspond to TESTDIR (MEMOS's parent).

Use of the double-period abbreviation is most common. Using the double periods you can quickly move up one level in the directory tree:

```
C:\TESTDIR\MEMOS> CD .. <ENTER>
C:\TESTDIR>
```

As you become more comfortable with directory operations, you may find these two abbreviations save you time and typing.

FASTOPEN Improves System Performance

Each time you access a file, whether to execute a program contained in an EXE or COM file or to open a word processing document, DOS must locate the file on your disk. As you've read, a directory is a list of filenames, sizes, and date and time stamps. In addition, the directory contains the location on disk where the file begins.

To find a file, DOS must first read the directory, which also resides on disk. DOS then searches the directory for the file entry. Although hard disks are fast, their mechanical nature makes them much slower than your computer's electronic components. Depending on the size of your directory, the read and search operation may take considerable time. If you are using DOS 3.3 or later, the FASTOPEN command can improve your system performance by reducing the amount of time DOS spends searching for the files you repeatedly use. Although your disk may contain thousands of files, you probably only use a small number of files on a regular basis—such as a word processing file, several key spreadsheets, or a set of database files. If you have a set of files you access often, FASTOPEN will improve your system performance.

FASTOPEN works by saving the name and location of each file you access. Assume, for example, you edit the file REPORT.TXT using your word processor. The first time you open the file, DOS will read and search the directory as discussed. When DOS locates the file, FASTOPEN will place the file's name and starting disk location in a list FASTOPEN stores in your computer's fast electronic memory. If you close the file to perform a different task and later reopen the file, DOS will first search FASTOPEN's list to see if the list contains the file's starting location. Because FASTOPEN's list is in your computer's fast electronic memory, this search only takes an instant. If FASTOPEN's list contains the file, DOS doesn't have to perform the slow directory-read-and-search operation.

FASTOPEN doesn't make every file open faster; FASTOPEN only improves performance when you are opening a file you recently accessed. However, because most users actually have a set of files they frequently use throughout the day, FASTOPEN will make a performance difference. If you are using DOS 3.3 or later on a hard disk, issue the following FASTOPEN command.

```
C:\> FASTOPEN  C:=50 <ENTER>
```

In this case, FASTOPEN will track the locations of up to 50 files. If you open more than 50 files throughout your computer session, FASTOPEN simply replaces older entries with more recently used files. FASTOPEN will track up to 999 files. However, because most users only repeatedly use a small number of files, tracking 50 files should be more than sufficient. In addition, each file FASTOPEN tracks consumes 32 to 48 bytes depending on your DOS version. If you create a large FASTOPEN list, you may simply waste memory. Because you can remove and insert floppy disks, FASTOPEN only supports hard disks.

If your computer has more than one hard disk, you can direct FASTOPEN to track files for both disks, as shown here:

```
C:\> FASTOPEN  C:=50  D:=50 <ENTER>
```

In this case FASTOPEN will track 50 files for drive C and 50 for drive D.

FASTOPEN uses a portion of your computer's available electronic memory to store its list and the software that searches the list for matching files. If you want to change the number of files FASTOPEN tracks, you must first restart your system. In Chapter 18, you will learn that most users place a FASTOPEN command in the special file AUTOEXEC.BAT so DOS automatically executes FASTOPEN each time your system starts. For more information on FASTOPEN, see the Command Reference at the end of this book.

SUMMARY

The TREE Command

The TREE command displays your disk's directory structure (or directory tree). To display all the directories on a disk, invoke TREE with the root directory symbol, as shown here:

```
C:\> TREE \ <ENTER>
```

If you include the /F switch, TREE will list the files in each directory. If you don't specify a directory name in TREE's command line, TREE displays only those directories that are below the current directory.

The PATH Command

The PATH command lets you display, change, or remove your command path. The command path is a list of directories DOS automatically searches when it fails to locate an external command in the current directory. To search the command path, DOS begins with the first directory specified, followed by the second, until DOS finds a matching command or exhausts its directory list. Your command path should only include directories containing your commonly used commands, such as the DOS directory. Once you identify your commonly used directories, place an appropriate PATH command in the batch file AUTOEXEC.BAT, discussed in Chapter 18. The following PATH command directs DOS to search drive C directories DOS and BATCH, in that order.

```
C:\> PATH C:\DOS;C:\BATCH <ENTER>
```

To display your current command path, invoke the PATH command with no parameters.

Abbreviating Directory Names

DOS uses directory path names as a road map to find your files and directories. Although you can always specify a complete directory path that begins at your disk's root directory, there may be times you can reduce

SUMMARY

the path names (and your typing). If the directory you desire resides within the current directory, you can access the directory by name without placing a backslash character before the name. For example, assuming the directory LETTERS resides within the current directory TESTDIR, the following RMDIR commands are the same.

```
C:\TESTDIR> RD  LETTERS <ENTER>

C:\TESTDIR> RD  \TESTDIR\LETTERS <ENTER>
```

If the directory is not within the current directory, you must specify a complete path name.

. and ..

Each time you create a directory, DOS places the entries . and .. in the directory. These two entries are actually abbreviations. The single period abbreviates the current directory, while the double periods abbreviate the parent directory (the directory immediately above the current directory in the directory tree). Every directory on your disk except the root has a parent. Using directory abbreviations may save time and keystrokes.

The FASTOPEN Command

If you are using DOS 3.3 or later, the FASTOPEN command may improve your system performance by reducing the amount of time DOS spends locating commonly used files on your hard disk. FASTOPEN doesn't reduce the amount of time for every instance you open a file, just the times for files you have recently used. The following FASTOPEN command directs FASTOPEN to track up to 50 files on drive C.

```
C:\> FASTOPEN  C:=50 <ENTER>
```

Hands On

Issue the following PATH command to display the current command path:

```
C:\> PATH <ENTER>
```

If your system does not have a command path defined, DOS will display the following message:

```
C:\> PATH <ENTER>
No path
```

Only if DOS displays the "No path" message should you execute the following PATH command:

```
C:\> PATH  C:\DOS
```

If your system has a command path defined, note the directories DOS searches to locate your external commands, as well as the order in which DOS searches the directories.

If necessary, use CHDIR to select the root directory as the current directory and invoke the TREE command.

```
C:\> TREE <ENTER>
```

Because your command path includes the DOS directory, DOS finds TREE although TREE.COM is in a different directory.

When TREE completes, invoke TREE including the /F switch to display the name of every file and directory on your disk. If the directory names scroll past you faster than you can read them, don't worry. You learned in Chapter 5 that you can use F3 to repeat the command and, when you press ENTER, be ready with CTRL-S to suspend screen scrolling and CTRL-S again to resume scrolling. In Chapter 20, you will learn how to use the MORE command to display a command's output a screenful at a time.

Next, use CHDIR to select the DOS directory.

```
C:\> CD  DOS <ENTER>
```

Because the DOS directory is immediately adjacent to the root, you did not have to include the backslash before the directory name.

Next, using the period abbreviation, list the files in the current directory.

```
C:\DOS> DIR  . <ENTER>
```

When the command completes, use the double periods to list the files in the parent directory.

```
C:\DOS>  DIR  .. <ENTER>
```

Using the double periods with CHDIR, move one level up the directory tree.

```
C:\> CHDIR  .. <ENTER>
```

Last, if you are using DOS 3.3 or later, issue the following FASTOPEN command.

```
C:\> FASTOPEN  C:=50 <ENTER>
```

Review

1. What is the command path?
2. What is the function of the following command?

```
C:\> PATH  C:\DOS;C:\BATCH  <ENTER>
```

3. How do the following commands differ?

```
C:\> PATH C:\DOS;C:\BATCH <ENTER>
C:\> PATH \DOS;\BATCH <ENTER>
```

4. How does the abbreviation . differ from .. ?
5. When can you reference a directory by name without specifying a complete path name?
6. How does FASTOPEN improve your system performances?

Answers

1. The command path is the list of directories DOS searches to locate your external DOS commands when it fails to locate a command in the current directory. The PATH command lets you define your command path.
2. The command places the directories DOS and BATCH into the command path. If DOS fails to locate an external command in the current directory, DOS will first search the directory DOS for the command. If DOS locates the command, DOS will execute it, and the search ends. If DOS fails to locate the command, it will then search the directory BATCH.
3. The first command ensures that DOS will look for the directories DOS and BATCH on drive C. The second command, however, directs DOS to look for the directories on the current drive. If drive C is the current drive, DOS will find the directories as desired. If a drive other than C is the default, DOS won't locate the directories, unless they also happen to be on the second disk.
4. The single period (.) is an abbreviation for the current directory. The double periods (..) are an abbreviation for the current directory's parent (the directory that appears immediately above the current directory in the directory tree).
5. If the desired directory resides within the current directory, you can access it by name, without specifying a complete directory path.
6. If you are using DOS 3.3 or later, FASTOPEN lets DOS track the disk locations of your commonly used files. By default, when you access a file, DOS must read the directory from disk and search the directory to locate the desired file entry. Using the directory entry, DOS knows where the file begins on disk. If you repeatedly open the same set of files, FASTOPEN will keep track of where the files begin on disk, storing the information in a list in your computer's fast electronic memory. When you access a file, DOS first searches FASTOPEN's list of recently opened files. If DOS finds the desired file, DOS does not have

to perform the slow directory-read-and-search. DOS will only find entries for recently opened files in FASTOPEN's list. Therefore, FASTOPEN improves the performance for files you open more than once during a user session, but not for every file you use.

Advanced Concepts

The PATH command lets you specify the list of directories in which you want DOS to search for your EXE, COM, and BAT files. DOS uses the command path when it fails to locate an external command. Using the command path, DOS lets you define default locations it should always search for commands. If you are using DOS version 3.2 or later, the APPEND command lets you define a data-file search path. Using APPEND, you can define a list of directories you want DOS to search each time it fails to locate a data file.

To better understand how APPEND works, create the following data file in the TESTDIR directory.

```
C:\TESTDIR> COPY  CON  UNLV.TXT <ENTER>
UNLV Runnin Rebel Basketball!
^Z <ENTER>
        1 File(s) copied
```

Next, select the root directory as the default and issue this TYPE command.

```
C:\> TYPE  UNLV.TXT <ENTER>
File not found
```

Because the data file is not in the current directory, TYPE cannot find the file, and the command fails. Using the following APPEND command, direct DOS to search the TESTDIR directory when it fails to locate a data file.

```
C:\> APPEND  C:\TESTDIR <ENTER>
```

Next, from the root directory, repeat the previous TYPE command.

```
C:\> TYPE  UNLV.TXT <ENTER>
UNLV Runnin Rebel Basketball!
```

Because DOS locates the file in the data-file search path, the command succeeds. DOS uses the data-file search path restrictively. DOS will not, for example, delete a file it finds using the data-file search path. For example, if you try to delete a file named UNLV.TXT from the root directory, DOS will not delete the corresponding file in the TESTDIR directory. Depending on your DOS version, DOS will display an "Access denied" or "File not found" message:

```
C:\> DEL  UNLV.TXT <ENTER>
File not found
```

Use of the APPEND command is much less common than PATH. However, if you have a program that only runs from a directory that contains specific files, you may be able to use APPEND to eliminate the directory restriction. For more information on APPEND, see the Command Reference at the end of this book.

Command-Path Tricks

The maximum number of characters your command path can contain is 121. Even though users should only include in their command path directories most likely to contain commonly used commands, users somehow manage to push the 121-character limit. Chapter 26 examines the SUBST command, which lets you substitute an unused disk drive letter for a directory name. For example, the following command abbreviates the DOS directory as drive E:

```
C:\> SUBST  E:  C:\DOS <ENTER>
```

After you substitute a drive letter for a directory name, you can use the drive letter as you would the directory name. For example, the following DIR command uses drive E to list the files in the DOS directory:

```
C:\> DIR  E: <ENTER>
```

If your command path has become excessively long, you may be able to reduce its length by substituting drive letters for the longer directory names. See Chapter 26 for more information on SUBST.

Deleting an Entire Directory Tree

Prior to DOS 6, the only way to delete a directory tree was to individually delete all of the files in each subdirectory and then to remove each subdirectory using RMDIR. To simplify this process, however, DOS 6 provides the DELTREE command that lets you delete all of the files and subdirectories in a directory tree in one step. Because an inadvertent DELTREE command can have devastating results, I have placed the DELTREE discussion in the "Advanced Concepts" section. The format of the DELTREE command is as follows:

```
DELTREE [/Y] [drive:]directory_path
```

The optional /Y switch directs DELTREE to perform the subdirectory deletion without first prompting you to verify the operation. The following command, for example, directs DELTREE to remove the directory OLD_DIR and all of the subdirectories that it contains.

```
C:\> DELTREE OLD_DIR <ENTER>

Delete directory "OLD_DIR" and all its subdirectories? [yn]
```

As you can see, by default, DELTREE displays a yes or no prompt asking you to verify the operation. If you type Y, DELTREE will remove the directory tree. If you instead type N, DELTREE will leave the directory unchanged. If you want DELTREE to suppress the yes or no prompt, include the /Y switch as shown here:

```
C:\> DELTREE /Y <ENTER>
```

The DELTREE command will delete all of the files that reside in the specified directory or files that reside in subdirectories contained within the directory. DELTREE will delete the files even if the read-only or hidden attributes are set.

NOTE

The DELTREE command supports wildcard characters to help you quickly delete specific directories. Be very careful before you issue a DELTREE command that uses wildcards—an inadvertent command could quickly erase many directories that you did not intend to remove.

Key Terms

Directory tree A name given to your disk's directory structure because the directories grow from the root directory much like the branches or root structure of a tree.

Command path The list of directories DOS searches to locate your external commands. The PATH command defines your command path.

Parent directory The directory above the current directory in the directory tree.

Data-file search path The list of directories DOS searches to locate your data files. The APPEND command defines your data-file search path.

Chapter 12

Fundamental Copy Operations

In Chapters 10 and 11 you created and used DOS directories to organize your files. Because of the importance of this subject and its frequency of use, this chapter examines the steps you must follow to copy files from a hard disk to a floppy disk, or vice versa. Next, the chapter presents the DISKCOPY command, which duplicates an entire disk's contents. Using DISKCOPY, you can make working copies of the original floppy disks that accompany the software programs you buy. Once you copy the originals, store them in a safe place.

Copying Files from a Hard Disk to a Floppy Disk

No matter what programs you run on your computer, word processor, spreadsheet, or database, eventually you will need to copy one or more files from your hard disk to a floppy, to exchange information with another user, or to keep as a backup. In fact, one of the most common questions DOS users ask is "What is the command to copy a file from a Lotus 1-2-3 directory to a floppy?"

You learned in Chapter 10 that when you use an application, such as Lotus 1-2-3 or WordPerfect, you will normally use CHDIR to change to the program's directory before you begin. If you want to copy a file from one of your applications, use CHDIR to go to the directory containing the file.

Next, place the floppy disk to which you want to copy the file in drive A. Use DIR to examine the floppy disk's current contents.

```
C:\123> DIR  A: <ENTER>
```

If DOS displays the following error message, you must format the floppy disk for use, as discussed in Chapter 7.

```
General failure reading drive A
Abort, Retry, Fail?
```

If the floppy disk contains files, make sure your copy operation won't overwrite an existing file whose contents you need. Also note the amount of free disk space on the floppy. If the available disk space is less than the size of the file you want to copy, you'll need to use a different floppy disk or delete the unwanted files before you begin copying.

If the disk has sufficient space, issue a COPY command that specifies your filename as the source file and drive A as the destination:

```
C:\123> COPY FILENAME.EXT  A: <ENTER>
```

If you are copying several files, use a DOS wildcard if one can selectively copy the desired files. Otherwise, you may have to issue a series of COPY commands, one for each file.

Copying from a Floppy Disk to a Hard Disk

Copying files from a floppy disk to your hard disk essentially follows the steps just presented. To begin, use CHDIR to select the directory to which you want to store the files.

Next, place in drive A the floppy disk containing the files to copy and issue the following DIR command:

```
C:\123> DIR  A: <ENTER>
```

Examine the files on the floppy disk closely to make sure you won't overwrite existing files on your hard disk. If you accidentally overwrite a file's contents, you can't get the information back.

If you want to copy all the files on the floppy disk, you can use a wildcard combination, as shown here:

```
C:\123> COPY  A:*.* <ENTER>
```

DOS will copy the files from drive A to the current directory of your hard disk using each file's original name.

If you only want to copy selected files from the floppy disk, use a wildcard, if appropriate, or issue individual COPY commands for each filename, as shown here:

```
C:\123> COPY  A: FILENAME.EXT <ENTER>
```

Whether you are copying files from your hard disk to floppy or vice versa, the copy operation becomes much easier if you select the correct directory before you begin.

DISKCOPY Copies an Entire Disk's Contents

One of the most important rules for working with your computer is never to use the original disks containing the software you buy. Instead, use the DISKCOPY command to duplicate the disk's contents, and place the original disk in a safe location.

DISKCOPY is an external DOS command in your DOS directory. Depending on the number and size of your floppy drives, the steps you will perform to copy a disk will differ slightly. Before you begin, however, use a write-protect tab, as discussed in Chapter 1, to protect the original floppy disks. If you are using 3 1/2-inch floppies, slide the write-protect cover into the write-protect position. As with COPY, discussed in Chapter 8, DISKCOPY refers to the disk you are copying as the source and the disk you are copying to as the target.

The DISKCOPY command completely overwrites the existing contents of the target disk. Make sure the target disk does not contain needed information. If the target disk is unformatted, DISKCOPY will format the disk for you.

You might remember from Chapter 7 that your disk stores information in sectors on your disk's circular tracks, as shown in Figure 12-1.

DISKCOPY copies a source disk to the target disk sector by sector, track by track. Therefore, DISKCOPY requires the target disk to be exactly the same type of disk as the source—you can't use DISKCOPY to copy a 5 1/4-inch disk to a 3 1/2-inch disk.

Two-Drive DISKCOPY

If your computer has two floppy disk drives of the same size, place the source disk in drive A and the target disk in drive B, as shown in Figure 12-2.

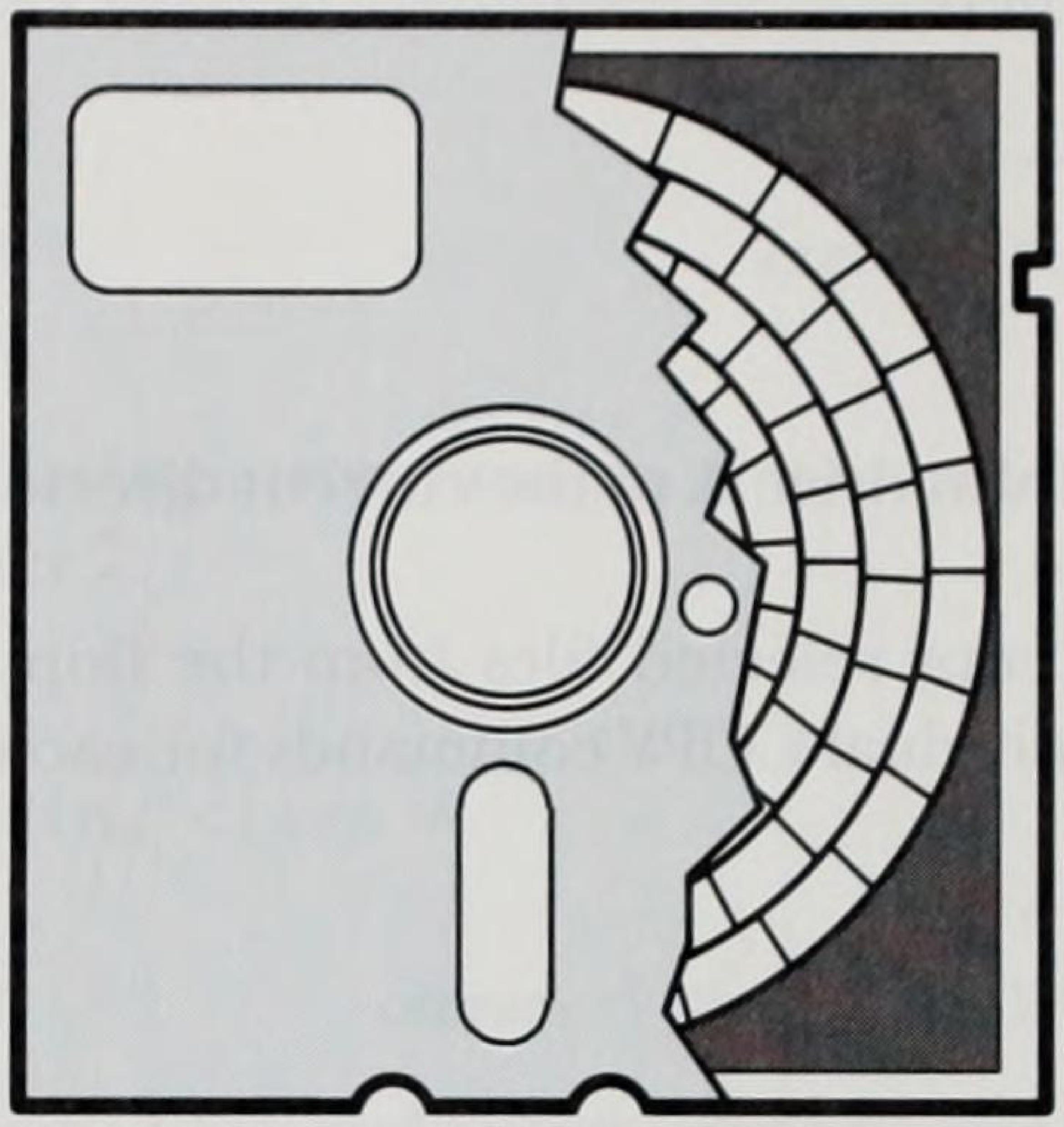
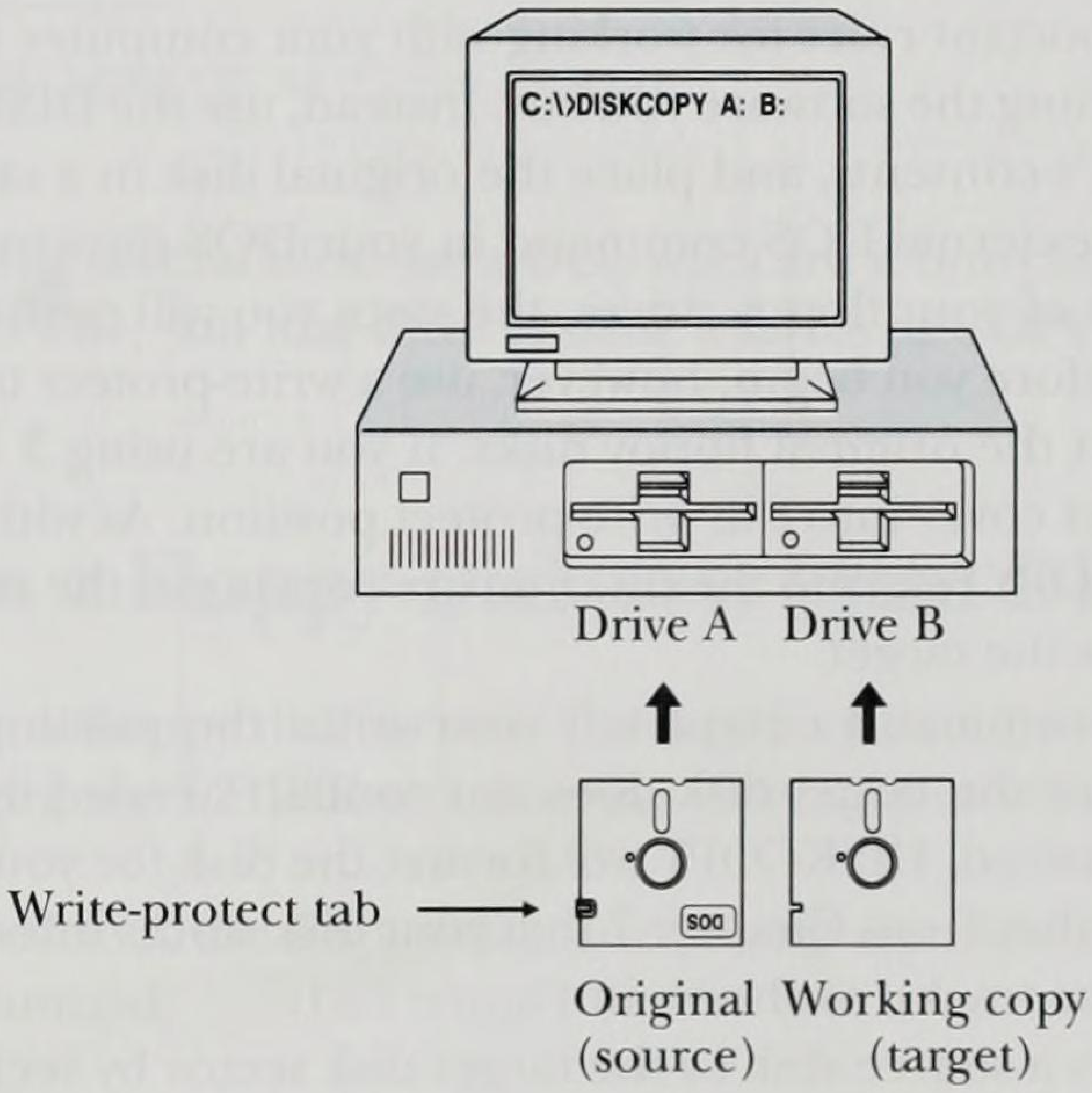


Figure 12-1. *DISKCOPY copies the source disk's tracks and sectors*



```
C:\>DISKCOPY A: B:
```

Drive A
Drive B
Write-protect tab
SOG
Original
(source)
Working copy
(target)

Figure 12-2. *For two-drive DISKCOPY, place the write-protected source disk in drive A and the target disk in drive B*

Next, issue the following DISKCOPY command:

```
C:\> DISKCOPY A: B: <ENTER>
```

DISKCOPY will display the following prompt to ensure that you have inserted the floppy disks in the drive.

```
C:\> DISKCOPY A: B: <ENTER>

Insert SOURCE diskette in drive A:

Insert TARGET diskette in drive B:

Press any key to continue. . .
```

When you press a key you will see drive A's disk activation light illuminate as DISKCOPY reads the disk's contents. Next, drive B's disk activation light will illuminate as DISKCOPY writes information to the disk.

If the disk in drive B is unformatted, DISKCOPY will format the disk displaying the following message.

```
Formatting while copying
```

When DISKCOPY completes, it asks if you want to copy a second disk:

```
Copy another diskette (Y/N)?
```

Type N, and DISKCOPY will end, returning control to the DOS prompt. Remove the target disk and label it. Don't write on a label attached to the disk—write on the label *before* you adhere it to the disk.

One-Drive DISKCOPY

If your computer only has one floppy disk drive of a given size, you will need to perform a single-drive DISKCOPY operation. To do so, you will place the (write-protected) source disk in the drive for DISKCOPY to read, and then replace it with the target disk when DISKCOPY is ready to write. Depending on your disk size, you may have to exchange disks two or more times. Figure 12-3 illustrates the process of exchanging the source and target disks during a single-drive DISKCOPY operation.

To perform a single-drive DISKCOPY, place the source disk in drive A and issue the following command:

```
C:\> DISKCOPY  A:  A: <ENTER>
```

DISKCOPY will display the following prompt to ensure that the source disk is in drive A.

```
Insert SOURCE diskette in drive A:

Press any key to continue. . .
```

When you press a key to continue, you will see drive A's disk activation light illuminate as DISKCOPY reads the disk's contents. When DISKCOPY is ready to write

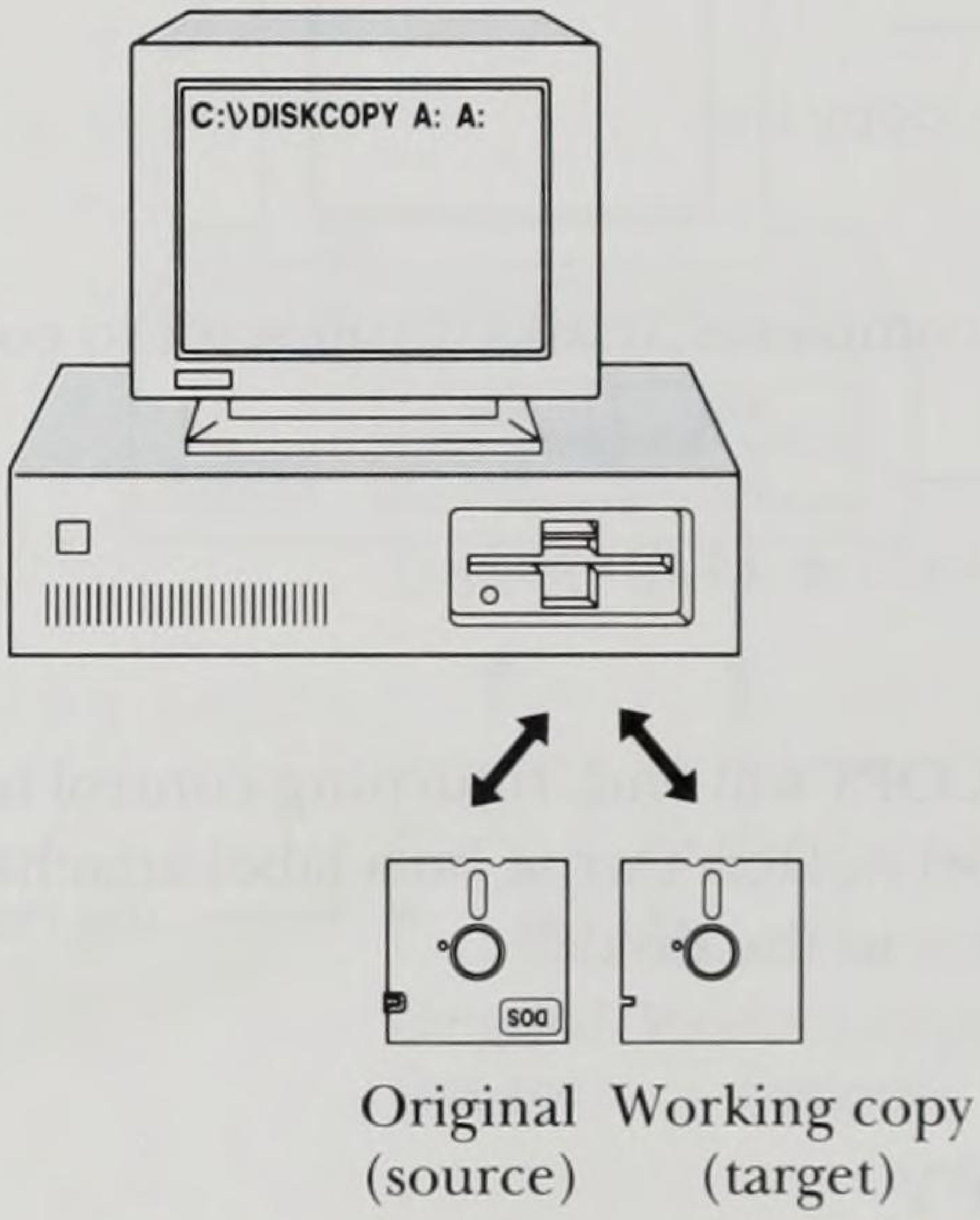


Figure 12-3. *In single-drive DISKCOPY, you must exchange the source and target disks in the same drive*

information to the target disk, DISKCOPY will display the following prompt for you to insert the target disk in the drive.

```
Insert TARGET diskette in drive A:

Press any key to continue. . .
```

Place the target disk in drive A and press any key to continue. As with double DISKCOPY, if the target disk is unformatted, DISKCOPY will format it. Depending on your disk size, DISKCOPY may prompt you to exchange the disks again. When DISKCOPY completes, it will ask if you want to copy another disk. As before, type N to end DISKCOPY.

SUMMARY

Copying Files

To copy a file from the current 123 directory in drive C to a floppy in drive A, you can use the following command:

```
C:\123> COPY FILENAME.EXT  A: <ENTER>
```

DOS will assume that the source file is in the current directory if a path isn't given.

Likewise, if you don't specify a destination disk, DOS assumes that you want the current drive and directory to be the destination. For example, the following command copies all of the files on the disk in drive A to the current directory, 123, in drive C:

```
C:\123> COPY  A:*.* <ENTER>
```

Whether you are copying files from your hard disk to floppy or vice versa, the copy operation becomes much easier if you select the correct directory before you begin.

SUMMARY

DISKCOPY

The DISKCOPY command copies all of a disk's files, track by track, sector by sector, to another disk of the same size and capacity. DISKCOPY will not copy files from a low-density to a high-density disk or from a 5 1/4-inch disk to a 3 1/2-inch disk.

If you have two identical drives (in this example, A and B), you can issue the following command to duplicate a disk without swapping disks:

```
C:\> DISKCOPY  A:  B: <ENTER>
```

After prompting you to put the original in drive A and the target in drive B, DOS will duplicate the disk.

If you have only one drive of a given type, you can make a duplicate by swapping the original and the destination disks in the same drive when you issue the following command:

```
C:\> DISKCOPY  A:  A: <ENTER>
```

You will then be prompted when to put in the source disk and when to put in the target disk.

DISKCOPY will format an unformatted target disk as part of the copy process.

Hands On

Insert a floppy disk in drive A. If the disk is unformatted, use the FORMAT command to prepare the floppy for use, as presented in Chapter 7. Use CHDIR to select the DOS directory as the current directory.

```
C:\> CD  \DOS <ENTER>
```

Next, use the following COPY commands to copy the files DISKCOPY.COM, FORMAT.COM, and LABEL.EXE to the disk in drive A.

```
C:\DOS> COPY DISKCOPY.COM  A: <ENTER>
        1 File(s) copied

C:\DOS> COPY FORMAT.COM  A: <ENTER>
        1 File(s) copied

C:\DOS> COPY LABEL.EXE  A: <ENTER>
        1 File(s) copied
```

Whether you are copying files from the DOS directory or any other directory, the process is the same. Select the directory containing the desired files and then copy the files by name, as shown.

If you are still using the original disks that accompanied any of your software packages, use the DISKCOPY command to make working copies of each disk. Place the original disks in a safe location.

Review

1. Why can't you use DISKCOPY to copy a 5 1/4-inch disk to a 3 1/2-inch floppy?
2. When must you perform a single-drive DISKCOPY?
3. What are the commands to copy the files BUDGET.DAT and SALARY.DAT from the directory 123 to the floppy disk in drive A?

Answers

1. The DISKCOPY command duplicates a disk by copying the source disk track by track, sector by sector to the target disk. Therefore, both disks must have the same number of tracks and sectors.
2. If your computer only has one floppy drive of any given size and capacity, you must perform a single-drive DISKCOPY.
3. The commands to copy the files BUDGET.DAT and SALARY.DAT from the directory 123 to the floppy disk in drive A are

```
C:\> CHDIR  \123 <ENTER>
C:\123> COPY  BUDGET.DAT  A: <ENTER>
C:\123> COPY  SALARY.DAT  A: <ENTER>
```

Advanced Concepts

One of the most frustrating problems users experience when copying files from a hard disk to a floppy occurs when the floppy disk does not have enough disk space to hold the files. When the floppy disk runs out of space, your screen will display the following message:

```
Insufficient disk space
```

The difficulty occurs in trying to resume the copy operation with the last file copied. If you have ever experienced this error, you know that COPY does not provide you with an easy way to resume the copy operation with the last uncopied file.

For cases where you need to copy more files than a single floppy disk can hold, use the ATTRIB and XCOPY commands. Chapter 23 discusses them in detail. The following provides you with a quick tip for copying a directory of files to floppy disk. To begin, select the directory containing the files you want to copy. Next, issue the following ATTRIB command to set each file's archive-required attribute:

```
C:\> ATTRIB  +A  *.* <ENTER>
```

After the ATTRIB command completes, issue the following XCOPY command, which copies each file whose archive-required attribute is set.

```
C:\> XCOPY *.*  A:  /M <ENTER>
```

Each time XCOPY successfully copies a file, XCOPY will clear the file's archive-required attribute. Eventually, the floppy disk will run out of space, and DOS will display the following message:

```
Insufficient disk space
```

Because XCOPY has been clearing the archive-required attribute for each file it successfully copies, you can easily continue the copy operation by inserting a new floppy disk in drive A and repeating the same XCOPY command.

```
C:\> XCOPY *.*  A:  /M <ENTER>
```

XCOPY will resume the copy operation with the first file whose archive-required attribute is set. If the floppy disk fills again, insert a new floppy disk and repeat the XCOPY command until all of the files are copied.

Key Terms

Source disk	The disk containing required information to copy from.
Target disk	The disk to which a DOS command will copy information.

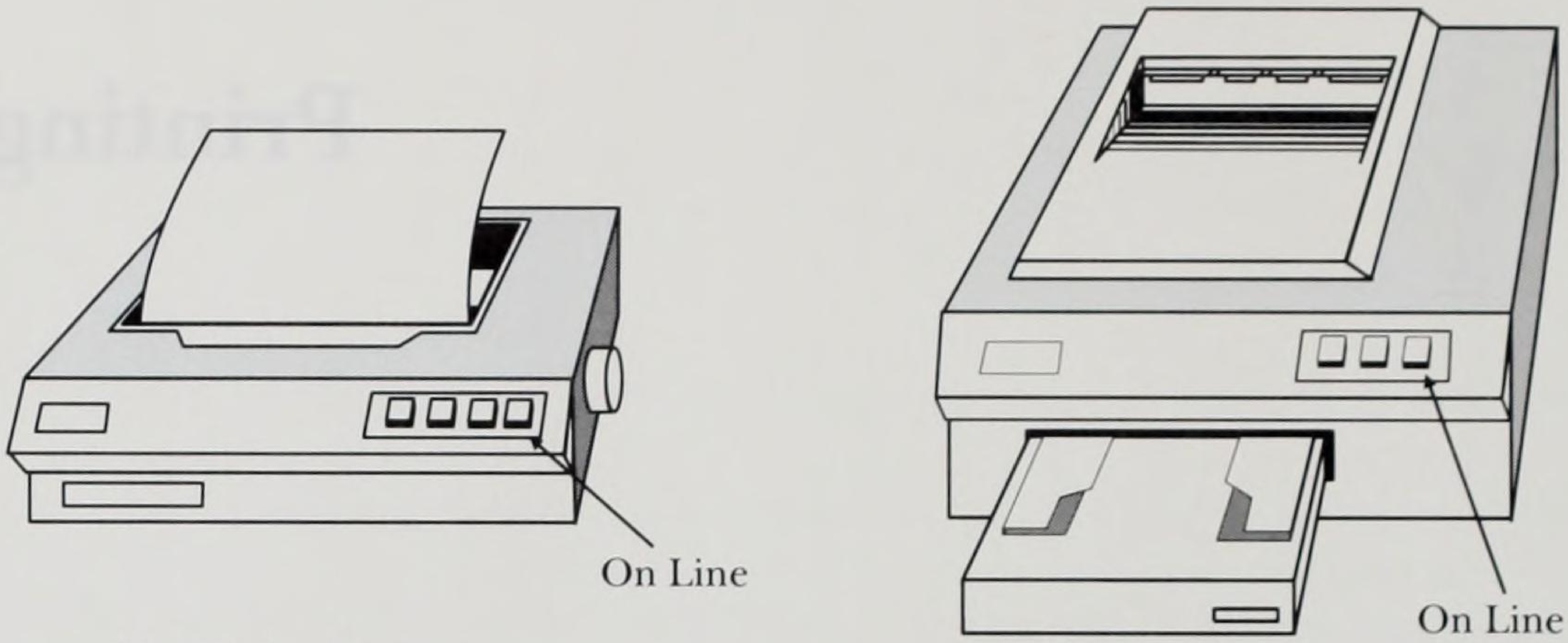
Chapter 13

Printing

No matter which applications (word processor, spreadsheet, or database) you use on your computer, you will eventually need to print information. Users call printed output *hard copy* because it is a physical copy of your information. In this chapter you will learn how to access your printer, not only from DOS, but from within your applications. In addition, you will examine several basic DOS printer capabilities, such as printing the contents of the current screen or writing a command's output to both the screen and printer at the same time.

Using Your Printer

Before you can use your printer, it must be attached to the back of your computer by a cable, plugged in, and turned on. In addition, most printers have a button labeled "On Line" or "ONLINE" or "Ready" that has a small light. When the light is on, your printer is "on the line," ready to accept data from the computer. When the light is off, the printer will ignore any information the computer sends it. The On Line button works like a toggle. Most printers begin on line when you first turn them on. Then the first time you press it, the printer goes off line. The next time you press it, the printer goes back on line. Figure 13-1 illustrates the common location of your printer's On Line button.



Tractor-feed printer

Laser printer

Figure 13-1. *Location of the printer's On Line button*

Identifying Your Printer Type

Whatever your printer brand, your printer connects to a port at the back of your computer with a cable. The cable type and port used will differ, depending on whether your printer is a *parallel* or *serial* printer. You must know your printer type (parallel or serial) because the steps you must follow to access your printer from DOS will differ.

Serial and parallel printers differ in the number of wires they use to send data to your printer. Briefly, serial printers receive their information from the computer on one line. Parallel printers, on the other hand, are receiving data on eight lines at the same time (in parallel). As you might expect, the parallel printers are the fastest. Since it has more data lines, a parallel connection needs more wires and pins than its serial counterpart. Don't be fooled by the number of pins in the plug, however; pins are often left unused. If you examine the cable that connects your printer to your computer, as well as the port on the back of your computer, you can determine if your printer is parallel or serial. Figure 13-2 shows what the ports on the back of your computer should look like.

Depending on your computer's hardware configuration, you may have from one to three parallel ports and one to four serial ports. Most systems, however, have at least one serial port and one parallel port. DOS assigns the names LPT1 through LPT3 to the parallel ports and COM1 through COM4 to the serial ports.

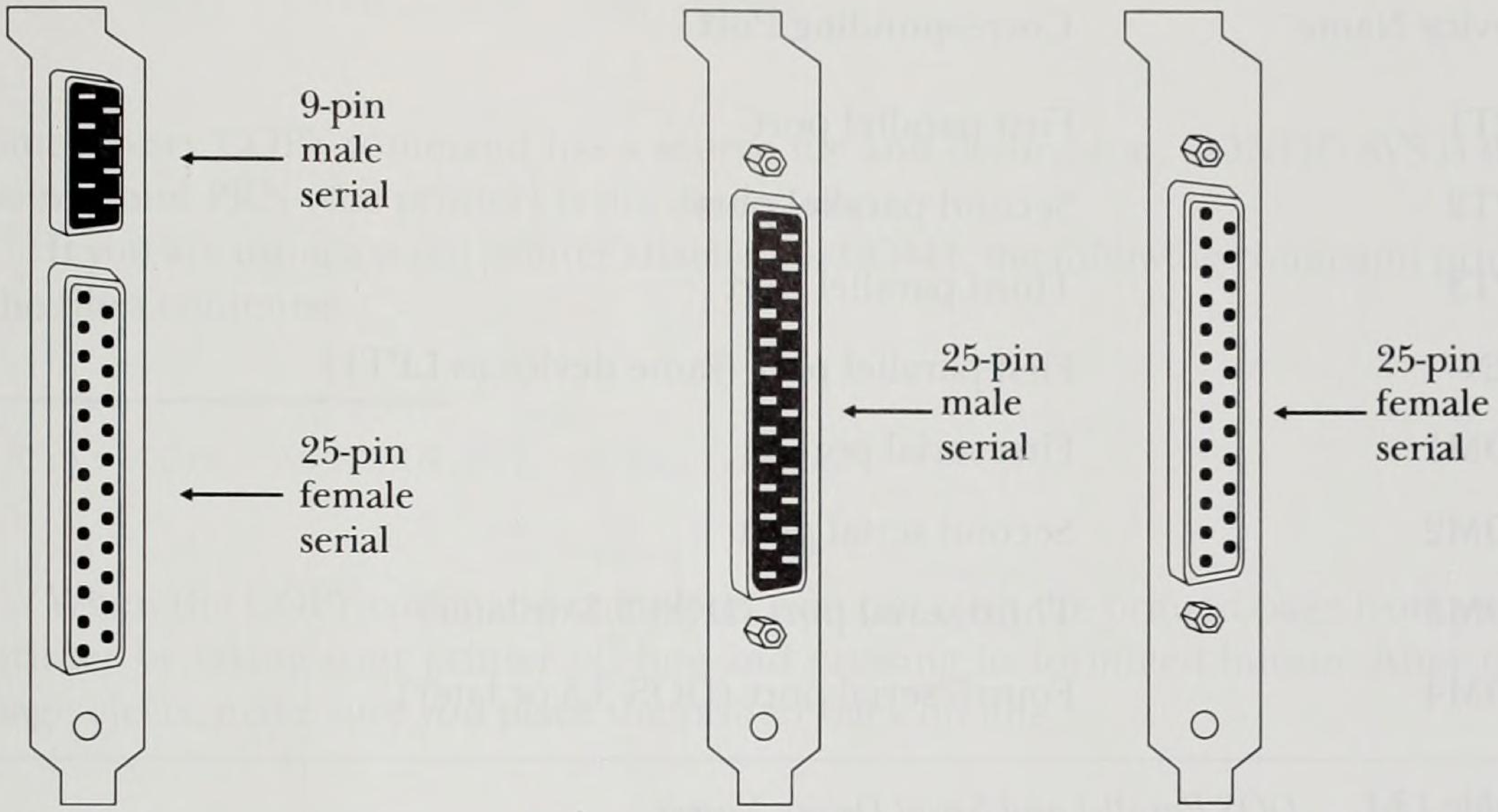


Figure 13-2. *Common parallel and serial printer ports*

The Device Names PRN and AUX

Most computers have at least one parallel and one serial port. DOS assigns the name LPT1 to the first parallel port and COM1 to the first serial port. Typically, the only devices users attach to the computer's parallel ports are printers, and hence the letters *LPT* (*line printer*) in the port name. Serial ports, however, connect printers, mice, and modems. Because serial ports were initially used for communications with either modems or unique devices, DOS uses the letters *COM* (*communications*) in the serial-port names.

In addition to the device names LPT*n*: and COM*n*: (the colon is optional), DOS lets you use PRN and AUX. Because most users have printers attached to the parallel port LPT1, DOS lets you refer to this port as *PRN*. Likewise, because the serial port COM1 normally connects auxiliary devices, DOS lets you refer to the port as *AUX*. Table 13-1 summarizes the DOS parallel and serial port names.

The Serial Port's Communication Settings

If your printer connects to a serial port, you have to make sure your printer and computer are using the same communication settings. If you are using a parallel printer, you don't have to worry about such matters.

Device Name	Corresponding Port
LPT1	First parallel port
LPT2	Second parallel port
LPT3	Third parallel port
PRN	First parallel port (same device as LPT1)
COM1	First serial port
COM2	Second serial port
COM3	Third serial port (DOS 3.3 or later)
COM4	Fourth serial port (DOS 3.3 or later)

Table 13-1. *DOS Parallel and Serial Device Names*

If when you try to use your serial printer, nothing is printed or the hardcopy output contains unrecognizable or nonsense characters, your serial port's communication settings don't match your printer's. In such cases, you must refer to the documentation that accompanied your printer to determine the proper communication settings. Next, you must use the MODE command to assign those settings to the serial port. For example, the following MODE command assigns the proper communication settings for an HP LaserJet Series printer to the serial port COM1.

```
C:\> MODE  COM1:96,N,8,1,P <ENTER>
```

The actual MODE command you must execute depends on your specific printer brand. Chapter 41 discusses the MODE command in detail.

Copying Files to Your Printer

In Chapter 8, you learned how to copy a file from one location on your disk to another. You can also copy files to your printer using the COPY command. If you are using a parallel printer attached to LPT1, you can use the following command to print the file CONFIG.SYS from your disk's root directory:

```
C:\> COPY  \CONFIG.SYS  PRN <ENTER>
```

Since every COPY command has a source file and destination, CONFIG.SYS is the source and PRN (the printer) is the destination.

If you are using a serial printer attached to COM1, the following command prints the file's contents:

```
C:\> COPY  \CONFIG.SYS  COM1: <ENTER>
```

When the COPY command completes, you can eject the printed page from your printer by taking your printer off line and pressing its formfeed button. After the page ejects, make sure you place the printer back on line.

Printing the Screen

Depending on your application, there are many times when it is convenient to print a copy of the screen's contents. For example, assume you are running an application that generates an error message you don't understand. By printing the screen's contents you can get a copy of the error message to use later as a reference when you discuss the error.

To print the screen's contents, press the SHIFT-PRTSC (print screen) key combination. When you do so, DOS will copy the current screen to your printer. As before, you will need to take your printer off line and then press the formfeed button to eject the page.

Printing the Screen with a Serial Printer

By default, when you press SHIFT-PRTSC, DOS sends its output to the parallel port PRN (LPT1 by default). If you are using a serial printer, you must execute the following MODE command to direct DOS to print the screen contents to your serial printer.

```
C:\> MODE  LPT1:=COM1: <ENTER>
```

When you do so, DOS will display the message in following example to indicate it is routing all output normally destined for LPT1 to COM1.

```
Resident portion of MODE loaded
LPT1: rerouted to COM1:
```

After you execute the MODE command, the SHIFT-PRTSC key combination will print the current screen using your serial printer. You only need to issue this MODE command once per user session. If you do this every session, you might consider putting this command into an AUTOEXEC.BAT file, so that it will execute automatically on startup (see Chapter 18 for more information).

Printing a Screen with Graphics

By default, the SHIFT-PRTSC key combination only supports screens of text, such as the output of the DIR command. If you want to print your screen contents when the screen contains a graphics image, such as a spreadsheet graph, and have it look like the screen does, you must first execute the GRAPHICS command.

The GRAPHICS command loads *memory-resident* software (software that remains in your computer's memory after the command completes) that helps DOS in printing a graphics image. GRAPHICS is an external DOS command that resides in the file GRAPHICS.COM.

The format of the GRAPHICS command is as follows:

```
GRAPHICS [printer_type][profile][/R][/B][/LCD][/PRINTBOX:id]
```

Items that appear within the left and right brackets are optional command-line entries. The *printer_type* option lets you specify your printer type. Table 13-2 lists the supported printer types.

The *profile* option specifies the complete directory path to a printer profile file that contains information about each of the supported printers listed in Table 13-2. If you don't specify a filename (and you probably won't), GRAPHICS uses the file GRAPHICS.PRO, which is in the DOS subdirectory.

The /R switch reverses the typical black-and-white color usage. By default, DOS prints white screen images as black and black screen images as white. The /R switch directs DOS to print black screen images as black, and white screen images as white.

The /B switch directs DOS to print the background images in color. This switch is only valid for the COLOR4 and COLOR8 printer types.

The /LCD option directs DOS to use the LCD screen aspect ratio. Many portable computers use LCD (liquid crystal display) screens.

The /PRINTBOX option selects the size of the print box. The two valid print box *ids* are STD (for standard size) and LCD (liquid crystal display).

Printer Type	Description
COLOR1	IBM color printer with a black ribbon
COLOR4	IBM color printer with a red, green, blue, and black ribbon
COLOR8	IBM color printer with a cyan, magenta, yellow, and black ribbon
GRAPHICS	IBM Personal Graphics Printer, Prowriter, or Quietwriter
GRAPHICSWIDE*	IBM Personal Graphics Printer with an 11-inch-wide carriage
HPDEFAULT*	Any Hewlett-Packard PCC printer
DESKJET*	Any Hewlett-Packard DeskJet printer
LASERJET	Hewlett-Packard LaserJet printer
LASERJETII*	Hewlett-Packard LaserJet Series II printer
PAINTJET*	Hewlett-Packard PaintJet printer
QUIETJET*	Hewlett-Packard QuietJet printer
QUIETJET PLUS*	Hewlett-Packard QuietJet Plus printer
RUGGEDWRITER*	Hewlett-Packard RuggedWriter printer
RUGGEDWRITER WIDE*	Hewlett-Packard RuggedWriter Wide printer
THERMAL	IBM Thermal printer
THINKJET*	Hewlett-Packard ThinkJet printer

Table 13-2. *Printer Types Supported by GRAPHICS.COM (*indicates DOS 5 or later printer support)*

The following command invokes GRAPHICS for a Hewlett-Packard LaserJet II printer:

```
C:\> GRAPHICS  LASERJETII  C:\DOS\GRAPHICS.PRO <ENTER>
```

The only time you must execute the GRAPHICS command is when you want to use the SHIFT-PRTSC key combination to print a screen containing graphics.

Sending Information to Both Screen and Printer

In some cases you might want DOS to send all the information it writes to the screen and printer. For example, you might want DOS to print a copy of your directory listing as it displays the filenames on your screen. The CTRL-PRTSC key combination directs DOS to do just that. To begin, make sure your printer is on line. Next, press the CTRL-PRTSC key combination and then issue the DIR command. As DIR displays the filenames on your screen, DOS also *echos* them to the printer, which prints them. When the DIR command completes, press CTRL-PRTSC a second time to turn off the printer echoing.

The CTRL-PRTSC key combination works as a toggle. The first time you press CTRL-PRTSC, DOS turns on printer echoing. The second time you press CTRL-PRTSC, DOS turns the printer echo off.

As before, if you are using a serial printer, you must first issue the following command before you can use CTRL-PRTSC:

```
C:\> MODE LPT1:=COM1: <ENTER>
```

Telling Your Applications About Your Printer

Just as you must tell DOS where and what your printer is, so must you tell your applications. Normally, the program (word processor, spreadsheet, or database) will ask you during its installation to identify your printer port. If you try to use your printer from within an application and nothing happens, make sure first that the printer operates properly from DOS. If not, you may need to specify the correct printer speeds for a serial printer using the MODE command. If you can use the printer from DOS, most applications have a setup or menu option that lets you specify your printer type and port. Refer to the documentation that accompanied the product for details.

Hands On

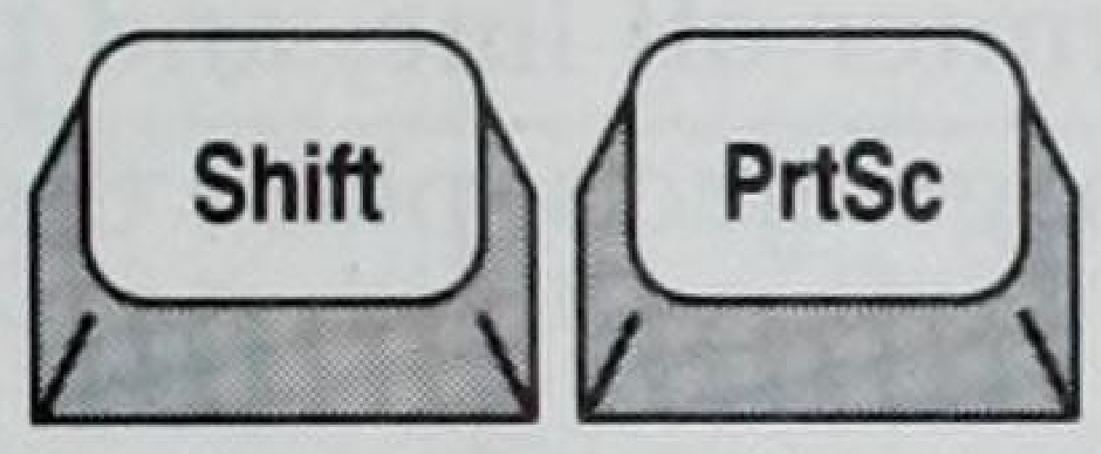
The following commands will differ slightly, depending on whether you are using a serial or parallel printer. If possible, examine your printer's cable to determine this.

SUMMARY

Parallel and Serial Printers

Depending on the number of wires a printer uses to receive information from the computer, your printer is either a parallel or serial printer. Your computer can have one to three parallel devices and up to four serial devices depending on your DOS version. Parallel printers are the fastest and the most common. DOS assigns the names LPT1 through LPT3 to the parallel printer ports. Serial printers, on the other hand, attach to the ports COM1 through COM4.

Printing the Screen

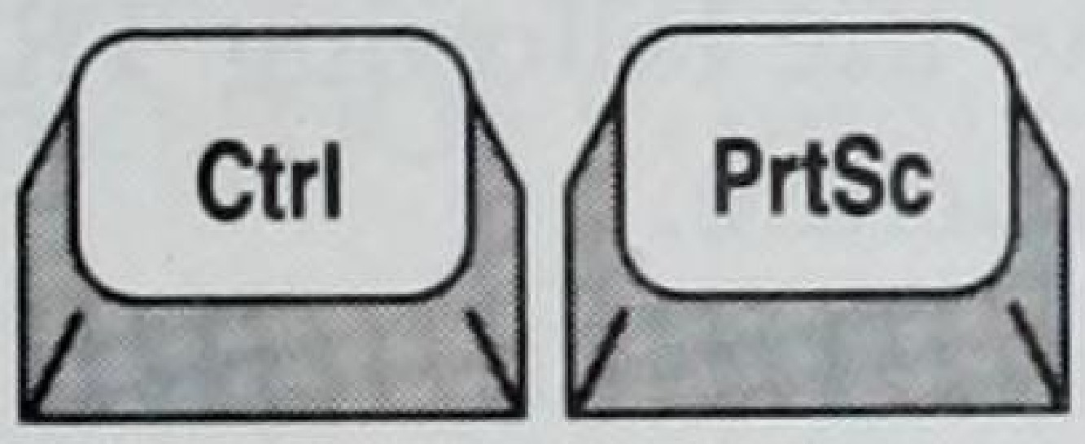


Press SHIFT-PRTSC to print the current screen contents. When DOS completes the screen copy, you can eject the page from your printer by taking your printer off line and pressing the formfeed button.

The GRAPHICS Command

By default, the SHIFT-PRTSC key combination correctly prints your screen contents when the screen contains only text data. To use the SHIFT-PRTSC key combination to print a graphics image, such as a spreadsheet graph, you must first issue the GRAPHICS command for your specific printer type.

Turning Printer Echo On and Off



The CTRL-PRTSC key combination toggles printer echoing on and off. When printer echoing is on, DOS sends each character it displays on the screen to the printer also. CTRL-PRTSC works as a toggle, meaning the first time you press CTRL-PRTSC, DOS enables printer echo. The second time you press CTRL-PRTSC, DOS turns printer echo off.

If you cannot tell, assume you have a parallel printer—most users do. If the printer commands fail to work, repeat the commands, following the steps for a serial printer.

To begin, copy the contents of the file AUTOEXEC.BAT, which resides in your root directory, to your printer using one of the following commands:

```
C:\> COPY  \AUTOEXEC.BAT  PRN <ENTER>
```

or

```
C:\> COPY  \AUTOEXEC.BAT  COM1: <ENTER>
```

If your serial printer prints meaningless characters, use the MODE command to define your port speed and settings as discussed earlier in this chapter.

When the COPY command completes, take your printer off line, and press the printer's formfeed button to eject the page. After the page ejects, place the printer back on line.

Next, if you are using a serial printer, issue the following MODE command:

```
C:\> MODE  LPT1:=COM1: <ENTER>
```

Next, issue the following DIR command to list the files in the current directory:

```
C:\> DIR  /W <ENTER>
```

When the DIR command completes, press SHIFT-PRTSC to print the current screen. When the operation completes, use the printer's formfeed button to eject the page.

Next, press CTRL-PRTSC to enable printer echo. Issue the following TREE command to print a copy of your disk's current directory structure:

```
C:\> TREE C:\ <ENTER>
```

When the TREE command completes, press CTRL-PRTSC once again to disable printer echoing. Issue the following DIR command:

```
C:\> DIR  \DOS\*.* <ENTER>
```

Because you have disabled printer echoing, DOS does not print the command's output.

Review

1. Why do parallel port names begin with LPT?
2. Why do serial port names begin with COM?
3. What is the PRN device?
4. How does SHIFT-PRTSC differ from CTRL-PRTSC?
5. Why must serial printer users issue the following MODE command before they can use SHIFT-PRTSC and CTRL-PRTSC?

```
C:\> MODE  LPT1:=COM1: <ENTER>
```

Answers

1. Parallel ports typically connect line printers to your computer. LPT is an abbreviation for line printer.
2. Serial ports commonly connect modems to your computer. A modem lets two computers communicate over telephone lines. COM is an abbreviation for communication.
3. PRN is another name for LPT1, the first parallel port in your computer.
4. SHIFT-PRTSC directs DOS to print the screen's current contents. CTRL-PRTSC directs DOS to print each line of text it writes to your screen.
5. By default, DOS sends printer output to the parallel port LPT1. If you want DOS to send the output to a serial port instead, you must use the MODE command.

Advanced Concepts

DOS lets you use one to three parallel ports (LPT1 through LPT3) and up to four serial ports (COM1 through COM4). When you buy a parallel or serial board, the manufacturer ships the board thinking it will be the first parallel or serial device in your computer (LPT1 or COM1). If your computer has more than one parallel or serial device, you must tell each board its device name. If you have two serial ports, for example, you must tell one board to respond to the device name COM1 and the other board to respond to the device name COM2. If you buy your computer with two or more ports already installed, your computer manufacturer has made sure each board knows its correct name. If you buy a second serial or parallel board and install the board yourself, you must tell the board its name.

WARNING

Never work on your computer while it is plugged in.

Depending on your board type, there are two ways to tell the board its name: DIP switches or jumpers. A DIP switch (dual inline package) is a collection of two or more very small switches on the board. The switches look like miniature light switches (see Figure 13-3).

The manual that accompanies your board will include a page of the correct DIP switch settings for each device name. Each switch normally has an Off or On (possibly 0 or 1) label. Using the point of a pin, a small screw driver, or a bent paper clip, you can change the DIP switch settings. Never change a board's DIP switch settings without first writing down the original values.

A *jumper,* on the other hand, is a small connector that connects two or more pins. Figure 13-4 illustrates a jumper.

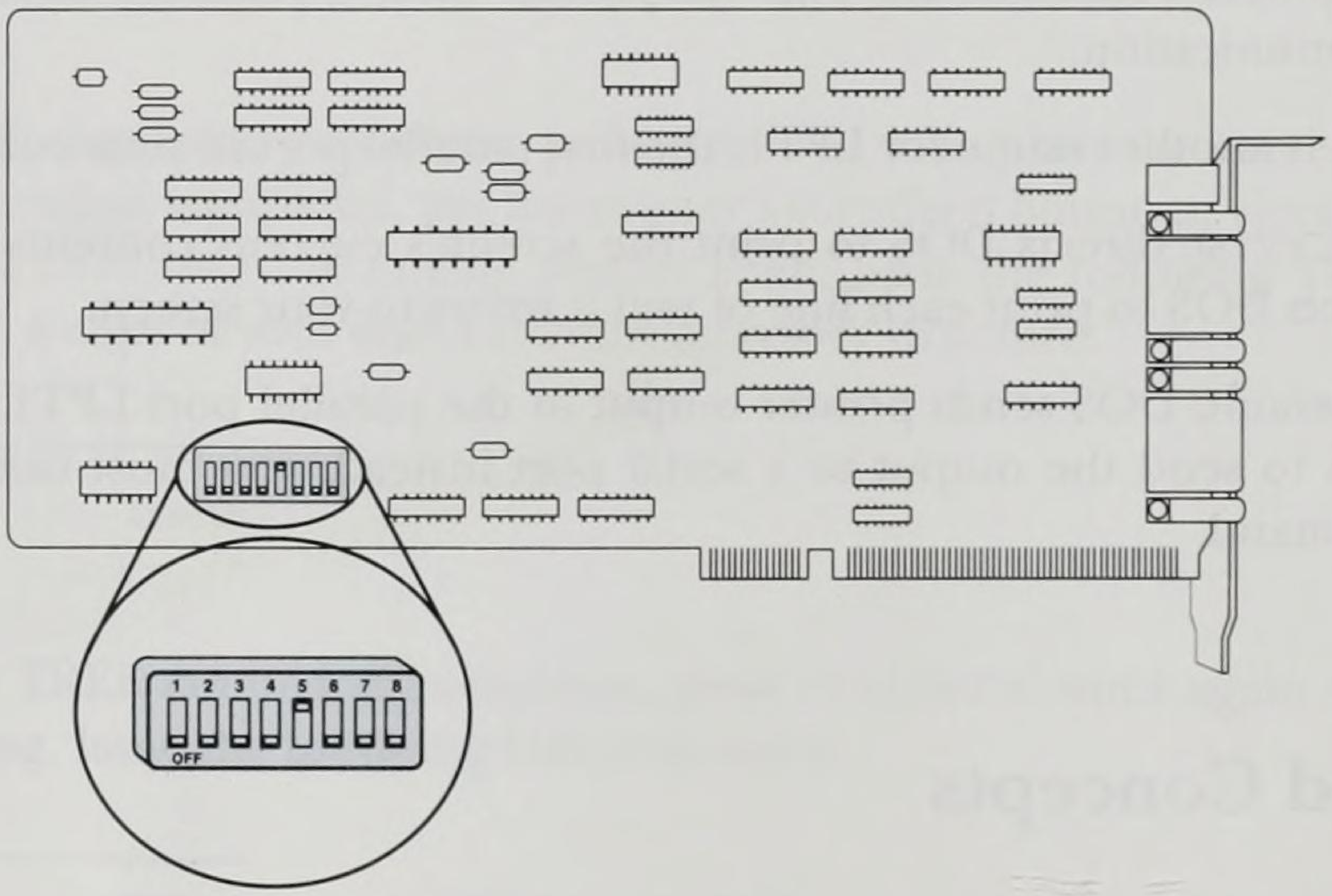


Figure 13-3. *DIP switches on a computer board*

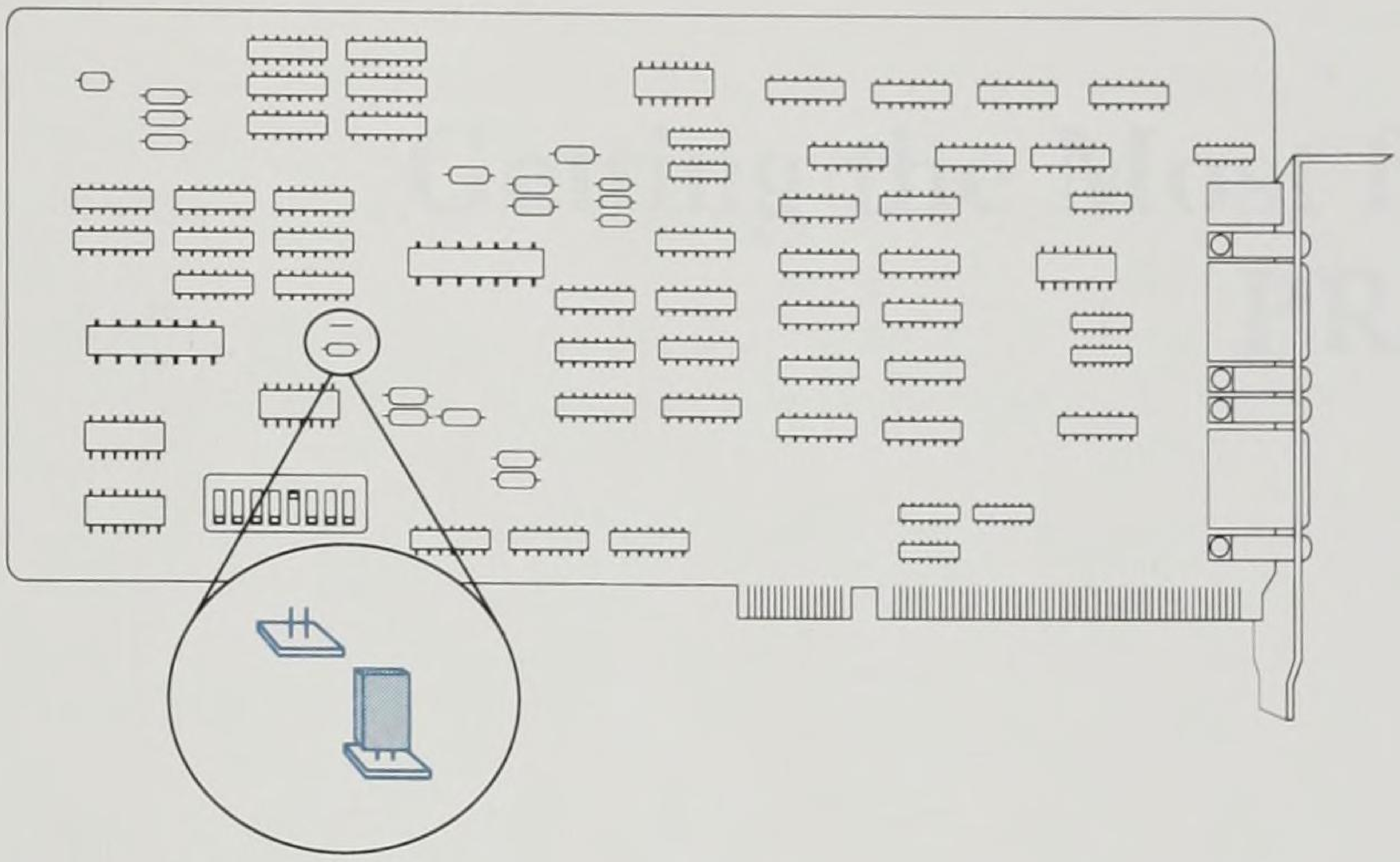


Figure 13-4. A jumper connects two or more pins

The manual that accompanied your board will contain jumper settings to select each device name. To change the jump, gently remove the jumper from the current pins. Next, carefully slide the jumper over the correct pins, taking care not to bend a pin. Again, never change a jumper before you note its original position. After you correctly set the board's switches or jumpers, close your computer's chassis, plug in your computer, and start DOS. Each board in your computer should now respond to its correct name.

Never change a board's DIP switch or jumper settings without first writing down the original values.

REMEMBER

Key Terms

Serial device A device such as a printer or mouse that connects to one of your computer's communication ports, such as COM1. Serial devices send or receive information from the computer one bit at a time over a single wire.

Parallel device A device, typically a printer, that connects to one of your computer's parallel ports, such as LPT1. Parallel devices communicate with the computer eight bits at a time over eight wires and, thus, are faster than serial devices.

Modem A hardware device that lets two computers communicate over standard telephone lines. Modem is an abbreviation for modulator/demodulator.

Chapter 14

Getting the Most from PRINT

In Chapter 13 you learned how to use basic DOS techniques to copy a file to your printer or to print your screen's current contents. If you have a word processor, you will normally want to use it to print your files due to the word processor's text formatting capabilities. There may be times, however, when you have a large text file, or several smaller files, that doesn't require special formatting. Then you can use the PRINT command to print the file's contents. Using PRINT, you can perform other DOS commands while the file prints, essentially letting your computer perform two tasks at once.

Understanding PRINT

In Chapter 13, you copied a file's contents to your printer using the COPY command and either the device name PRN or COM1, as shown here:

```
C:\> COPY \AUTOEXEC.BAT PRN <ENTER>
```

Although using COPY in this way can successfully print a file's contents, it has two shortcomings. First, in order to eject a page when COPY is completed, you must manually take your printer off line, press the printer's formfeed button, and place the printer back on line. Second, if you are copying a large file to the printer, the operation can take considerable time, leaving you unable to issue other commands until COPY completes.

To make it easier for you to print a large text file, or several smaller files, DOS provides the PRINT command. PRINT is an external DOS command in the DOS directory.

To print a text file using PRINT, invoke PRINT with the desired filename, for example:

```
C:\> PRINT  \AUTOEXEC.BAT <ENTER>
```

PRINT does not know if you are using a parallel printer attached to the port LPT1 (also called PRN) or a serial printer attached to COM1. Therefore, the first time you invoke it, PRINT will display the following prompt, asking you which *list device* to use.

```
Name of list device [PRN]:
```

The [PRN] tells you PRINT's default device. If you simply press ENTER, PRINT will use PRN (LPT1). If your printer attaches to a device other than PRN, such as COM1, you must type in the correct device name and press ENTER. If you are not sure to which device name your printer connects, simply press ENTER selecting PRN. If your printer fails to print the file, you will need to restart DOS and invoke PRINT using a different device name. Most users, however, will use parallel printers connected to PRN. PRINT only displays this device name prompt the first time you invoke it. After that, PRINT uses the device name specified until you restart DOS. If it cannot locate the file specified, PRINT will display "File not found." Otherwise, PRINT will begin printing the file's contents, displaying the following message and returning control to the DOS prompt.

```
C:\AUTOEXEC.BAT is currently being printed
```

After PRINT finishes printing the file's contents, PRINT automatically ejects the last page of the file for you.

Although it's not obvious for small files whose contents PRINT can print quickly, PRINT actually returns control to the DOS prompt before it finishes printing the file's contents, unlike COPY. PRINT is unique among all of the other DOS commands in that PRINT lets DOS redisplay its prompt for other commands before it has finished its processing. While you execute other commands, PRINT continues printing your files until it is done.

Printing More Than One File

PRINT is useful not only for printing a large text file, but also for printing several smaller files. To print multiple files, you can invoke PRINT several times, once for each file, or you can specify all the filenames in one PRINT command. The following PRINT command, for example, prints the contents of the root directory text files AUTOEXEC.BAT and CONFIG.SYS.

```
C:\> PRINT  \AUTOEXEC.BAT  \CONFIG.SYS <ENTER>
```

PRINT will begin printing the file AUTOEXEC.BAT, displaying the following message:

```
C:\AUTOEXEC.BAT is currently being printed
C:\CONFIG.SYS is in queue
```

Because you told PRINT the desired list device the first time you invoked it, PRINT does not prompt you to do so now.

As you can imagine, your printer is only capable of printing one file at a time. PRINT, therefore, prints files in the same order you specify. In this case, PRINT begins printing AUTOEXEC.BAT first. If you direct PRINT to print several files, PRINT will start printing the first file and will place the remaining filenames into a waiting list called a *queue.* As PRINT finishes one file, it begins printing the next file in the queue in a first-come, first-served basis. By default, PRINT lets you place up to ten files in the queue. If you need to queue more than ten files, you will need to invoke PRINT with the /Q switch as discussed in the "Advanced Concepts" section.

Assuming your directory contains the files FIRST.DAT, SECOND.DAT, and THIRD.DAT, you can add the files to the print queue, as shown here:

```
C:\> PRINT  FIRST.DAT  SECOND.DAT  THIRD.DAT <ENTER>

   C:\AUTOEXEC.BAT is currently being printed
   C:\CONFIG.SYS is in queue
```

```
  C:\FIRST.DAT is in queue
  C:\SECOND.DAT is in queue
  C:\THIRD.DAT is in queue
```

You learned in Chapter 9 that most DOS commands support the * and ? wildcards. PRINT is no exception. Using the * wildcard, the following command would print the files in the current directory with the BAT extension.

```
C:\> PRINT *.BAT <ENTER>
```

If you invoke PRINT without specifying a file, PRINT will display the name of the file it is now printing as well as the files in the queue:

```
C:\> PRINT <ENTER>

  C:\AUTOEXEC.BAT is currently being printed
  C:\CONFIG.SYS is in queue
  C:\FIRST.DAT is in queue
  C:\SECOND.DAT is in queue
  C:\THIRD.DAT is in queue
```

Canceling a File's Printing

PRINT quickly returns control to DOS before finishing the actual print job, letting you execute other commands. Because PRINT continues to work in the background while you execute other commands, PRINT ignores the CTRL-C keyboard combination that you usually use to end other DOS commands. If you want to cancel a file's printing, you must invoke PRINT again, using the /T or /C switches.

The /T switch directs PRINT to end the printing of the current file and to remove all files from the print queue. Assume, for example, your print queue contains the following files:

```
C:\> PRINT <ENTER>

  C:\AUTOEXEC.BAT is currently being printed
```

```
 C:\CONFIG.SYS is in queue
 C:\FIRST.DAT is in queue
 C:\SECOND.DAT is in queue
 C:\THIRD.DAT is in queue
```

You can cancel the file printing by invoking PRINT using /T:

```
C:\> PRINT /T <ENTER>
PRINT queue is empty
```

In addition, this line will print on your hard copy:

```
All files canceled by operator.
```

You should note that to improve their performance, most printers have their own electronic memories, called *buffers,* to which PRINT can send all or part of a file. If you use the /T switch to end a print job, your printer may still print several pages, depending on the size of its memory. PRINT, however, will not send any further information to the printer.

PRINT's /T switch directs PRINT to cancel the printing of all files. In many cases, you may simply want to remove a specific file from the print queue. PRINT's /C switch lets you do just that.

Assume, for example, your print queue contains the files ONE, TWO, THREE, FOUR, and FIVE:

```
C:\> PRINT <ENTER>

 C:\ONE is currently being printed
 C:\TWO is in queue
 C:\THREE is in queue
 C:\FOUR is in queue
 C:\FIVE is in queue
```

Using the /C switch, the following command removes the file THREE from the print queue:

```
C:\> PRINT THREE /C <ENTER>
```

PRINT will remove the file, displaying the queue's new contents:

```
C:\ONE is currently being printed
C:\TWO is in queue
C:\FOUR is in queue
C:\FIVE is in queue
```

When you use the /C switch, PRINT cancels the printing for the file whose name precedes the switch (including the one currently being printed), as well as those that follow it. The following command, for example, removes the files TWO and FOUR:

```
C:\> PRINT  TWO  /C  FOUR <ENTER>

 C:\ONE is currently being printed
 C:\FIVE is in queue
```

SUMMARY

PRINT

The PRINT command lets you print one or more text files, while still using your computer to perform other commands. When you direct PRINT to print several files, PRINT will begin printing the first file while placing the remaining files in a waiting list called a queue. Each time PRINT finishes printing a file, PRINT automatically ejects the file's last page from the printer and begins printing the next file specified in the queue. PRINT, like the TYPE command, only supports text files. The following command prints a copy of the file AUTOEXEC.BAT.

```
C:\> PRINT  \AUTOEXEC.BAT <ENTER>
```

Hands On

So that you have several files to print, use the COPY command to create the files BASEBALL.DAT, FOOTBALL.DAT, and HOCKEY.DAT, which contain the following team names.

```
C:\> COPY  CON  BASEBALL.DAT <ENTER>
Red Sox    <ENTER>
Giants     <ENTER>
White Sox  <ENTER>
Cubs       <ENTER>
Blue Jays  <ENTER>
^Z <ENTER>
        1 File(s) copied

C:\> COPY  CON  FOOTBALL.DAT <ENTER>
Seahawks <ENTER>
Raiders  <ENTER>
Giants   <ENTER>
Bills    <ENTER>
^Z <ENTER>
        1 File(s) copied

C:\> COPY  CON  HOCKEY.DAT <ENTER>
Rangers <ENTER>
Kings   <ENTER>
Black Hawks <ENTER>
Maple Leafs <ENTER>
^Z <ENTER>
        1 File(s) copied
```

Next, use the following command to print the file BASEBALL.DAT:

```
C:\> PRINT BASEBALL.DAT <ENTER>
```

PRINT, will only prompt you for the printer device the *first* time it is invoked:

```
Name of list device [PRN]:
```

If you are using a parallel printer connected to PRN, press ENTER. If your printer attaches to a different port, such as COM1, type in the port name and press ENTER.

Next, print the other two files:

```
C:\> PRINT  FOOTBALL.DAT  HOCKEY.DAT <ENTER>
```

While PRINT is printing your files, issue the following DIR command:

```
C:\> DIR <ENTER>
```

As you can see, PRINT lets you issue other DOS commands while it does its work in the background.

Using the asterisk wildcard, print all the files with the DAT extension.

```
C:\> PRINT  *.DAT <ENTER>
```

Next, using the PRINT command's /C switch, remove the file named HOCKEY.DAT from the print queue.

```
C:\> PRINT  HOCKEY.DAT  /C <ENTER>
```

Last, using the /T switch, remove all the files from the print queue and cancel the current print job. Examine the last page of the printout for the "All files canceled" message.

Review

1. How does the PRINT command differ from simply copying files to your printer using the following command?

```
C:\> COPY  FILENAME.EXT  PRN <ENTER>
```

2. What is the PRINT command to print the two root directory files AUTOEXEC.BAT and CONFIG.SYS?
3. What does the [PRN] in the following PRINT's prompt for a list device mean?

```
Name of list device  [PRN]:
```

4. What is the print queue?
5. Assuming your print queue contains the following files, what is the PRINT command to cancel the file LARGE.DAT?

```
C:\> PRINT <ENTER>

    C:\SMALL is currently being printed
    C:\MEDIUM is in queue
    C:\LARGE is in queue
```

Answers

1. Unlike COPY, the PRINT command returns control to the DOS prompt before it completes its processing. PRINT then shares control of the CPU with DOS, printing your file in the background while you are free to execute other commands. Second, when the PRINT command finishes printing a file's contents, it automatically ejects the file's last page from your printer.
2. The PRINT command to print the root directory files AUTOEXEC.BAT and CONFIG.SYS is

```
C:\> PRINT  \AUTOEXEC.BAT  \CONFIG.SYS <ENTER>
```

3. Letters appearing between left and right brackets normally indicate the default setting. In this case, the letters PRN tell you that if you simply press ENTER, PRINT will use the parallel port PRN. If your printer is attached to a port other than PRN, such as COM1, type in the correct device name and press ENTER.

4. A queue is a waiting list. A print queue is a list of files that are waiting to print. PRINT prints files in the same order they enter the queue. In other words, PRINT uses first-come, first-served order.
5. The proper command to cancel printing of LARGE.DAT is

```
C:\> PRINT  LARGE.DAT  /C <ENTER>
```

Advanced Concepts

If you use the PRINT command on a regular basis, PRINT provides several additional command-line switches you can use to improve performance. As you will learn in this section, you can only use these switches the very first time you invoke PRINT.

PRINT Installs Memory-Resident Software

Before DOS can run a program, the program must reside in your computer's electronic memory. In most cases, when you execute a command, DOS loads the command from disk and runs it. When the command completes, DOS makes the memory available for the next command.

Some DOS commands, however, don't run to completion in this manner. Instead, these commands remain in your computer's memory after DOS redisplays its prompt. In this way, the commands can periodically run to perform a specific task. Commands that remain in memory in this way are called *memory-resident commands*. Because such commands remain resident in memory after they return control to DOS, users often refer to them as *terminate-and-stay-resident* commands, or simply *TSR*s.

The PRINT command is a memory-resident command, which explains why PRINT can continue to print your files in the background while DOS executes other commands. After you invoke PRINT, you can visualize your computer's memory as having portions allocated to DOS and PRINT, as well as a portion available for other commands you will execute, as in Figure 14-1.

The switches discussed in the following section affect the way PRINT configures its memory-resident software. You can only use these switches the very first time you invoke PRINT—after that they "stay resident."

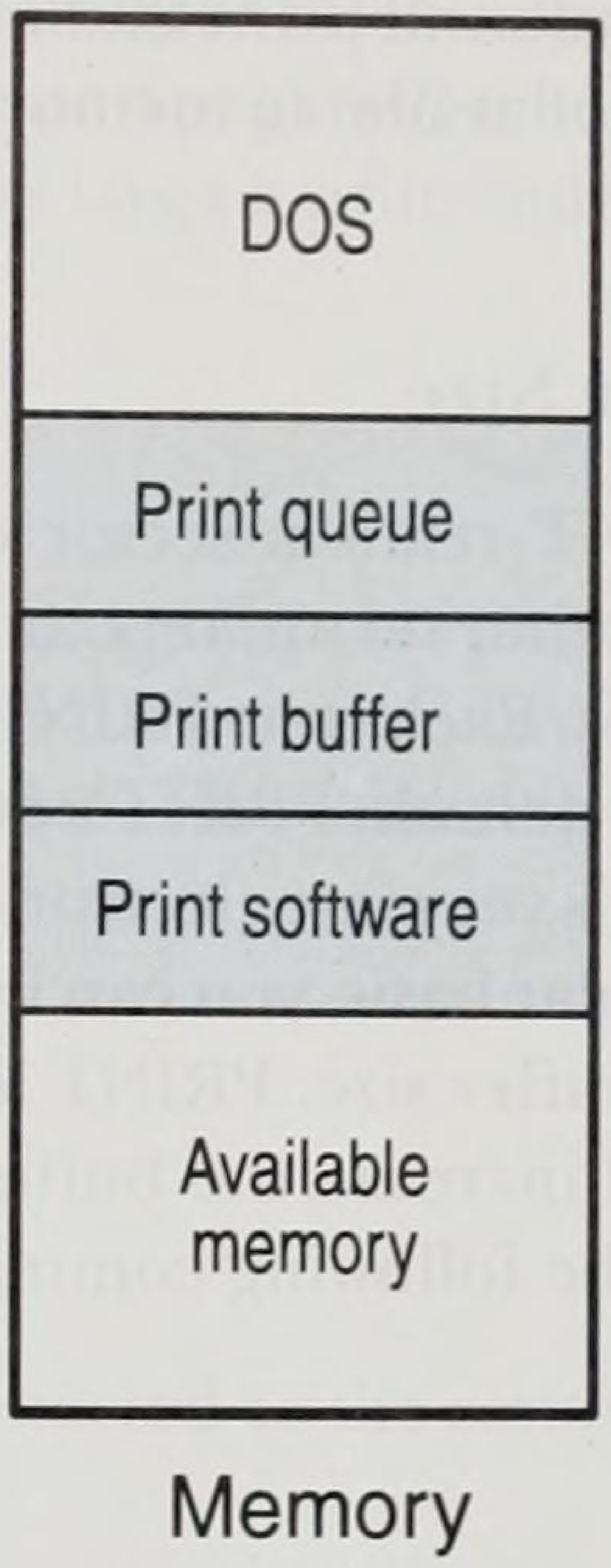

DOS
Print queue
Print buffer
Print software
Available memory
Memory

Figure 14-1. *PRINT is a memory-resident command residing in a portion of your computer's electronic memory*

/Q Increases PRINT's Queue Size

By default, PRINT lets you place up to ten files in the print queue. If you try to print more than ten files at one time, PRINT will display a message indicating the print queue is full.

PRINT's /Q switch lets you increase or decrease the print queue size. To set the queue size, specify a queue size from 4 to 32 after /Q. The following command, then, selects a queue size of 32:

```
C:\> PRINT /Q:32 <ENTER>
```

Use a colon to separate the desired queue size from /Q. Don't arbitrarily use the queue size 32 if you don't think you will print that many files on a regular basis. Each queue entry consumes 64 bytes of available memory.

/B Increases PRINT's Buffer Size

To print a file's contents, PRINT reads a sector of the file from disk, prints the information, and then reads another sector, repeating this process for the entire file. Figure 14-2 illustrates this process. Each time PRINT reads a sector, PRINT places the sector into a memory buffer. By default, PRINT's buffer size is 512 bytes.

Compared to your computer's very fast electronic memory, your hard disk is very slow. If you use PRINT on a regular basis, you can improve PRINT's performance by increasing PRINT's electronic buffer size. PRINT lets you specify a buffer size from 512 to 16,384 bytes. You should increase the buffer size in multiples of 512 (1024, 1536, 2048, 4096, and so on). The following command, for example, directs PRINT to use a 4096-byte (4Kb) buffer.

```
C:\> PRINT  /B:4096 <ENTER>
```

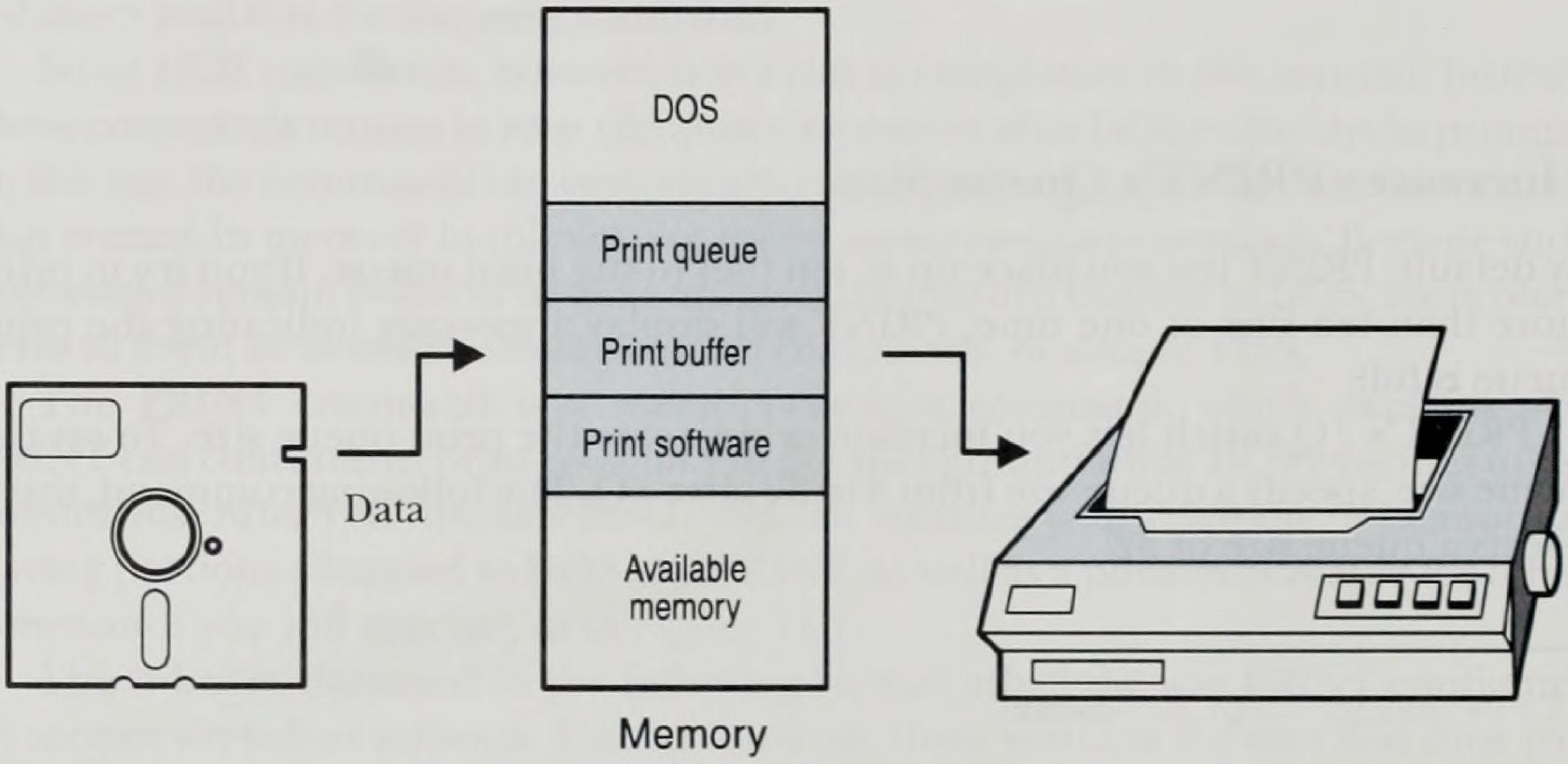


Figure 14-2. *PRINT reads a file's contents to a memory buffer, then prints it*

By increasing the buffer size, you increase the amount of information PRINT can read from your disk, and, in turn, reduce the number of slow disk operations PRINT must perform. A single-spaced, typed page will take about 4000 bytes. Use this value as your guide to determining how large a print buffer you need.

/D Specifies the List Device

The first time you invoke PRINT, it will prompt you for a list device. PRINT's /D switch lets you specify the list device when you invoke PRINT. Then, PRINT will not prompt you for the list-device name. The following command directs PRINT to use the device PRN.

```
C:\> PRINT  /D:PRN <ENTER>
```

If you are using a printer connected to the serial port COM1, use /D:COM1.

Giving PRINT More CPU Time

As you have read, PRINT is a memory-resident program that works in the background to print your files. When you use PRINT and issue other commands, your computer appears to perform two tasks at once. As it turns out, however, your computer is really letting PRINT work for a while, then your DOS command, then PRINT, and so on. Because your computer exchanges processing time between PRINT and your other commands so fast, the two commands appear to work at the same time.

Your computer achieves this illusion of running two commands at once by giving PRINT and DOS fixed time intervals (or *time slices*) in which to perform their work. This time interval is defined in terms of *CPU clock ticks,* which occur 18.2 times per second. In other words, the time interval lasts about 5/100 second. PRINT provides three switches that affect how often PRINT gets a turn and how long PRINT's turn lasts.

PRINT's /S switch determines how often PRINT gets to run. By default, PRINT does not get to run as often as the other DOS commands it is sharing the CPU with. The /S switch specifies the maximum number of clock ticks for which the other application can run before PRINT gets a chance. The default value for /S is 8 (4/10 of a second). You can specify values from 1 to 255 (1/20 second to 12 3/4 seconds). If you make the /S value too large, PRINT will not run as often—your files will still print, but they will take longer.

PRINT's /M switch specifies how long PRINT can keep the CPU each time PRINT gets a chance to run. The default value for /M is 2 (1/10 second). Using /M, you can give PRINT 1 to 255 clock ticks (1/20 second to 12 3/4 seconds). If you make the /M value too large, your other commands may become sluggish when PRINT is running.

Last, PRINT's /U switch specifies how long PRINT can wait for your printer if the printer isn't ready for more characters. Many older printers did not have very large built-in memory buffers to which PRINT could send characters. It was common for PRINT to be ready to send characters, but for the printer not to be ready to receive them. By default, PRINT will wait one clock tick for the printer to become available. If the printer remains busy, PRINT gives up its turn. PRINT's /U switch lets you increase the number of clock cycles PRINT can wait for the printer. For newer printers with large built-in memories, the /U switch normally isn't necessary. However, for older printers you might experiment with values of 8 or 16.

Placing a PRINT Command in AUTOEXEC.BAT

If you plan to make extensive use of PRINT, determine the switch settings that seem to give you the best performance. Next, place the corresponding PRINT command in your AUTOEXEC.BAT file so you don't have to worry about the switch details. Many users will place a command similar to the following in AUTOEXEC.BAT:

```
PRINT  /Q:16  /U:8  /M:32  /B:4096
```

Adding Files to the Print Queue

PRINT's /C switch lets you remove files from the print queue. When PRINT encounters the /C switch, PRINT cancels the file that precedes /C as well as all files that follow. In a similar manner, the /P switch directs PRINT to add a file to the queue. The /P switch exists to let you add one or more files to the print queue in a command line that cancels one or more files using /C. For example, the following command cancels the files OLDONE and OLDTWO while adding the files NEWONE and NEWTWO.

```
C:\> PRINT  OLDONE  /C  OLDTWO  NEWONE /P  NEWTWO <ENTER>
```

When PRINT encounters /P, PRINT will add to the queue the filename that precedes /P and all files that follow.

How Much Memory Does PRINT Allocate?

PRINT is a memory-resident program that allocates a portion of your computer's memory. Depending on your PRINT command-line switches, the amount of memory PRINT consumes will differ. To determine just how much memory your PRINT command uses, invoke either CHKDSK or the DOS 4 or later MEM command before you invoke PRINT and note the amount of free memory. Next, invoke your desired PRINT command. Repeat CHKDSK or MEM and note the amount of free memory. The difference between the two amounts is the amount of memory PRINT consumes.

Key Terms

Print queue A list of files waiting to print.

List device The printer device to which PRINT will send your hardcopy output.

Memory-resident software A program that remains in your computer's electronic memory after returning control to the DOS prompt. Memory-resident software is also called terminate-and-stay-resident or TSR software. PRINT is a memory-resident software program.

Chapter 15

Grouping Commands with Batch Files

Throughout Chapters 3 through 14, to execute a DOS command you type the command's name at the DOS prompt and press ENTER. When the command ends, DOS redisplays its prompt and waits for you to type the next command. Because you are in constant interaction with DOS, this process of executing a command, waiting for the command to complete, and executing the next command is called *interactive processing*. Interactive processing makes up the majority of the processing new DOS users perform. As you will learn in this chapter, you could save time and keystrokes by using DOS batch files.

A *batch file* is a file you create with your word processor or the DOS editor that contains the names of a "batch" of DOS commands to execute with one call. Batch files use the BAT extension. If you type the name of a batch file at the DOS prompt and press ENTER, DOS will execute, in order, each of the commands the batch file contains. Because DOS automatically executes the batch file's commands on its own, you are free to work on other tasks away from your computer, as opposed to waiting at the keyboard just so you can type in the next command.

Creating Simple Batch Files

DOS provides batch files to make your computer easier to use. You might create a batch file to abbreviate a long or difficult command, for example, or to execute several commands you always execute as a group. In this chapter you will create several very small batch files. As you learn more about DOS batch capabilities in Chapters 16 and 17, your batch files may grow from a few commands to very large collections, containing 10, 20, or even 50 or more commands. DOS batch files are very flexible and powerful tools. Many experienced DOS users are only now learning their true power.

Because the batch files presented in this chapter are quite short, you may find it easier and faster to copy the batch files directly from the keyboard. Before you create the batch files presented in this chapter, create a directory on your disk named BATCH, in which you can organize the files. Then make BATCH the current directory:

```
C:\> MKDIR  \BATCH <ENTER>
C:\> CHDIR  \BATCH <ENTER>
```

Next, the following COPY command creates a batch file named DATETIME.BAT that invokes the DATE and TIME commands.

```
C:\BATCH> COPY  CON  DATETIME.BAT <ENTER>
DATE <ENTER>
TIME <ENTER>
^Z <ENTER>
        1 file(s) copied

C:\BATCH>
```

From the DOS prompt, you can execute the batch file by typing its name:

```
C:\BATCH> DATETIME <ENTER>
```

When you execute a batch file, DOS executes, in order, each command the batch file contains. In this case, DOS will first execute DATE, displaying the following:

```
C:\BATCH> DATETIME <ENTER>

C:\BATCH> DATE
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy):
```

If your system date is correct, press ENTER to leave the date unchanged; otherwise, type in the correct date. When the DATE command completes, DOS will execute the TIME command:

```
C:\BATCH> DATETIME <ENTER>

C:\BATCH> DATE
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): <ENTER>

C:\BATCH> TIME
Current time is 11:16:23.67p
Enter new time:
```

Once again, type in the correct time or press ENTER to leave the current time unchanged. When the TIME command completes, the batch file has no more commands, and DOS redisplays its prompt.

Using an editor, edit the file DATETIME.BAT and insert a CLS command to clear the screen as soon as the batch file begins.

```
CLS
DATE
TIME
```

Again, from the DOS prompt, invoke DATETIME.BAT. As before, DOS will execute, in order, each of the commands the batch file contains.

DOS batch files exist to reduce your keystrokes and to eliminate the need for you to memorize long or difficult commands. Assume, for example, you typically use your computer to perform word processing in WordPerfect 5.1. Each time your system starts (if you've established no command path, as discussed in Chapter 11), you use CHDIR to select the WP51 directory, and you then invoke WordPerfect using the WP command, as shown here:

```
C:\> CHDIR  WP51 <ENTER>
C:\WP>  WP <ENTER>
```

Using a DOS batch file named DOWP.BAT, for example, you can automatically execute both commands in only one step. In this case, DOWP.BAT would contain the following commands:

```
CHDIR  \WP51
WP
```

To invoke the batch file, you would simply type **DOWP** at the DOS prompt.

```
C:\BATCH> DOWP <ENTER>
```

If you are trying to help new users get started, you can eliminate many confusing intermediate-step commands such as CHDIR with DOS batch files.

Naming Batch Files

Like all your files, you should assign meaningful names to your batch files. A batch file's name should summarize the task the batch file performs. The previous examples used the meaningful names DATETIME and DOWP. Never give your batch files the same name as a DOS internal command. If you do, DOS will always execute the internal command, never getting to your batch file. If you give your batch file the name of an external DOS command, such as FORMAT or DISKCOPY, DOS will execute either the batch file or the external command, depending on which one DOS encounters first. In general, assigning a batch file the name of an external command is asking for confusion.

Disabling Command-Name Display

Invoke the batch file DATETIME.BAT and examine your screen. By default, when DOS executes a batch file command, DOS displays, or echos, the DOS prompt and the command name on your screen, as shown here:

```
C:\BATCH> DATETIME <ENTER>

C:\BATCH> CLS          ←———————————————— Screen clears after this
C:\BATCH> DATE        ←————————————————— DATE command name
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): <ENTER>

C:\BATCH> TIME      ←——————————————————— TIME command name
Current time is 11:16:23.67p
Enter new time:
```

As your batch files become larger and more complex, displaying the command names in this way adds unnecessary screen clutter, which may confuse or intimidate a new user. By placing the ECHO OFF command as the first command in your batch file, you can disable command-name display, as shown here:

```
ECHO OFF
CLS
DATE
TIME
```

When you invoke DATETIME now, your screen will display the following:

```
C:\BATCH> DATETIME <ENTER>

C:\BATCH> ECHO OFF
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): <ENTER>
Current time is 11:16:23.67p
Enter new time: <ENTER>
```

You may not see all this, because your screen will clear immediately after the ECHO OFF command executes. As you can see, DOS did not display the names of the DATE and TIME commands.

However, DOS does display the command ECHO OFF. If you are using DOS version 3.3 or later, precede ECHO OFF with the @ ("at" sign) character, as shown here:

```
@ECHO OFF
CLS
DATE
TIME
```

The @ directs DOS to suppress the display of the current command's name.

Ending a Batch File

As you have learned, the CTRL-C keyboard combination lets you end DOS commands. If you press CTRL-C during a batch file's execution, DOS doesn't know if you want to end the current command or if you want to end the entire batch file. As a result, DOS displays the following prompt to find out:

```
Terminate batch job (Y/N)?
```

If you type **Y**, DOS will end the batch file, redisplaying the DOS prompt. If you instead type **N**, DOS will end the current command, continuing the batch file's execution with the next command in the batch file.

For example, invoke the DATETIME batch file. When DATE prompts you to enter a date, press CTRL-C. As discussed, DOS will prompt you to determine if you want to end the batch file or the current command:

```
C:\BATCH> DATETIME <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): ^C

Terminate batch job (Y/N)?
```

Type **Y** to end the batch file. DOS will end the batch file, and will redisplay the DOS prompt.

```
C:\BATCH> DATETIME <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): ^C

Terminate batch job (Y/N)?Y

C:\BATCH>
```

Repeat this process, pressing CTRL-C at the DATE prompt. This time, respond to the DOS prompt to end the batch file by typing **N** to end the DATE command, but not the entire batch file. DOS will continue the batch file's execution with the TIME command, as shown here:

```
C:\BATCH> DATETIME <ENTER>
Current date is Mon 04-05-1993
Enter new date (mm-dd-yy): ^C

Terminate batch job (Y/N)?N
Current time is 11:16:23.67p
Enter new time:
```

Using PATH to Locate Commonly Used Batch Files

To help you organize your batch files, you should create a directory named BATCH. Place into the directory your most commonly used batch files. Next, using the PATH command discussed in Chapter 11, include the BATCH directory in your command path. In the simplest case, your path might become the following:

```
C:\> PATH  C:\DOS;C:\BATCH <ENTER>
```

Display the current path before you change it to ensure that you include all the necessary directories. As discussed in Chapter 18, the special batch file AUTOEXEC.BAT lets you specify a list of commands you want DOS to automatically execute each time your system starts. By including a complete PATH command in AUTOEXEC.BAT, you can simplify your command-path definition.

Three Commonly Used Batch Commands

Almost every batch file you will encounter uses the ECHO OFF command to disable command-name display during the batch file's execution. In addition to suppressing the display of command names, ECHO also lets you display messages to the user as your batch file executes. The following batch file, GREETING.BAT, uses ECHO to display the message "Hi, have a nice day!"

```
@ECHO OFF
ECHO Hi, have a nice day!
```

If you are using a DOS version earlier than 3.3, replace @ECHO OFF with an ECHO OFF and a CLS command. When you execute GREETING.BAT from the DOS prompt, your screen will display the following:

```
C:\BATCH> GREETING <ENTER>
Hi, have a nice day!
```

The ECHO command serves two purposes. First, using ECHO OFF you disable command-name display. If you specify text other than OFF (or ON, which turns command-name display back on), ECHO will display the text as a message on your screen. As you increase your batch file knowledge, your batch files may eventually use ECHO to display menus such as the following:

```
          Menu

   F1 Start WordPerfect
   F2 Start Lotus 123
   F3 Run dBASE IV
   F4 Quit to DOS
```

Placing Remarks in Batch Files

Your batch filename should summarize the processing the batch file performs. Even so, you should develop the habit of placing remarks in your batch files that better explain the task the batch file performs and why. If another user reads your batch file, the remarks will help the user understand each step the batch file performs. If you read the batch file yourself, the remarks serve to remind you why you created the batch file, along with the steps the batch file performs.

DOS provides the REM command to let you place remarks in your batch files. Using REM, you should include, at a minimum, your name, the date you created the batch file, and the function the batch file performs. The following batch file uses REM to describe the DOWP.BAT.

```
@ECHO OFF
REM DOWP.BAT - By Kris Jamsa 2-26-93
REM Function:  Let the user invoke WordPerfect in one step
CHDIR \WP51
WP
```

When it executes your batch files, DOS ignores the lines containing REM, knowing the lines contain user documentation. Make sure you place your REM commands after the ECHO OFF command in your batch file, or DOS will display each command, cluttering your screen.

When your batch files are very short, you may wonder why the batch file needs remarks. If your batch files only contain two or three common commands such as CLS, DATE, and TIME, the batch file won't need a lot of explanation. You should still include lines describing who created the batch file, when, and why. When your batch files get longer and more complex, remarks are essential to your understanding of the batch file's processing. It only takes a few minutes to properly document your batch files. Those few minutes can save you a great deal of time when you later try to recall the steps the batch file performs.

Temporarily Suspending a Batch File

As your batch file complexity grows, you may need to perform a specific action, such as placing a floppy disk in a drive or turning on the printer, before the batch file can perform the next task. The PAUSE command temporarily suspends a batch file until the user presses a key to continue. PAUSE displays an optional message followed by a prompt for the user to press any key to continue, as shown here:

```
[optional message]
Press any key to continue. . .
```

The optional message typically contains instructions for the user to perform a specific task. For example:

```
PAUSE Make sure your printer is on
```

When DOS encounters the above PAUSE command within a batch file, the screen will display the following:

```
PAUSE Make sure your printer is on
Press any key to continue. . .
```

When the user presses a key, DOS will continue the batch file's execution with the next command. In a different application, the PAUSE command might display a warning message, stating the batch file is about to delete files or perform some other critical operation. If the user doesn't want the batch file to continue, the user can press CTRL-C and end the batch file.

There is one problem with the PAUSE command; PAUSE does not work well with batch files using ECHO OFF. As discussed, the ECHO OFF command suppresses command-name display. Unfortunately, ECHO OFF also suppresses the PAUSE command's optional message. The following batch file, BATCOPY.BAT, for example, copies the batch files residing in the current directory to a floppy disk in drive A. The batch file intends to use the PAUSE command to prompt the user to place a formatted disk in drive A.

```
@ECHO OFF
PAUSE Place a formatted disk in drive A
COPY  *.BAT  A:
```

What happens, however, is that the ECHO OFF command suppresses the PAUSE command's optional message. When you invoke BATCOPY, your screen will display the following:

```
C:\BATCH> BATCOPY
Press any key to continue. . .
```

Because ECHO OFF suppresses the message telling the user to place a disk in drive A, most users will simply press any key to continue, and the COPY command will fail. In this case, the easiest way to correct the problem is to remove the ECHO OFF command.

```
PAUSE Place a formatted disk in drive A
COPY  *.BAT  A:
```

Many times, however, a PAUSE command will appear in the middle of your batch file. The easiest way to correct the problem here is to use ECHO to display the desired message and PAUSE simply to prompt the user to press a key:

```
@ECHO OFF
ECHO  Place a formatted disk in drive A
PAUSE
COPY  *.BAT  A:
```

When you invoke this batch file, your screen will display the correct instructions.

```
C:\BATCH> BATCOPY <ENTER>
Place a formatted disk in drive A
Press any key to continue. . .
```

SUMMARY

Batch Files

A DOS batch file is a file with the BAT extension that contains the name of one or more DOS commands. Like DOS commands, you execute DOS batch files by typing the batch file's name at the DOS prompt. When it encounters a batch file, DOS automatically executes, in order, the commands whose names the batch file contains. DOS batch files exist to save you time, keystrokes, and to eliminate the need to memorize long or difficult commands.

ECHO OFF

By default, DOS displays the DOS prompt and command name for each batch file command it executes. Using the ECHO OFF command, you can disable command-name display. Place the ECHO OFF command as the first command in your batch file. If you are using DOS 3.3 or later, precede ECHO with an "at" sign (@), @ECHO OFF.

```
ECHO OFF                                   @ECHO OFF
CLS                                        DATE
DATE                                       TIME
TIME
```

(DOS 3.2 and earlier)   (DOS 3.3 and later)

SUMMARY

Canceling a Batch File

As DOS lets you use CTRL-C to end a command, DOS also lets you cancel a batch file using CTRL-C. When you press CTRL-C during a batch file's execution, DOS doesn't know if you want to end the current command or if you want to cancel the entire batch file, so it displays the following prompt:

```
Terminate batch file (Y/N)?
```

If you type Y, DOS will cancel the batch file, redisplaying the DOS prompt. If you type N, DOS will end the current command, continuing the batch file's execution with the command that immediately follows.

The ECHO Command

The ECHO command lets your batch files enable or disable command-name display using ECHO OFF or ECHO ON. In addition, ECHO lets your batch files display messages to the user. Using the ECHO command, you can create menu-driven batch files, or even set your screen colors and key combinations, as discussed in Chapter 34.

The REM Command

The REM command lets you include explanatory remarks in your batch files that explain who wrote the batch file, when the batch file was written, and the function the batch file performs. REM is an abbreviation for remark. When DOS encounters a REM command in your batch file, it ignores the command and continues the batch file's execution with the next command. Select a format that you like that explains who created the batch file, when, and why, and include the information after ECHO OFF in your batch files, for example:

```
@ECHO OFF
REM Written by:
REM Date written:
REM Function:
```

The PAUSE Command

The PAUSE command displays an optional message to the user and then suspends your batch file's processing until the user presses any key to con-

SUMMARY

tinue. Using PAUSE, your batch file can display instructions or status messages to the user, pausing for a user response. If the user doesn't want the batch file to continue, the user can press CTRL-C to end the batch file. The following PAUSE command directs the user to place the printer on line.

```
PAUSE  Place your printer on line
```

When DOS encounters the PAUSE command in your batch file, your screen will display the following.

```
PAUSE  Place your printer on line
Press any key to continue. . .
```

Hands On

If you have not yet done so, use the MKDIR command to create a BATCH subdirectory. Select BATCH as the current directory.

```
C:\> MD  \BATCH <ENTER>
C:\> CD  \BATCH <ENTER>
```

Next, create the batch file WIDEDIR.BAT that clears your screen and then display your directory listing in a wide format using DIR's /W switch.

```
CLS
DIR /W
```

Invoke WIDEDIR from the DOS prompt, as shown here:

```
C:\BATCH> WIDEDIR <ENTER>
```

Next, create the batch file HELLO.BAT that displays a personalized greeting.

```
@ECHO OFF
ECHO Hi, NAME
ECHO How are you today?
```

Replace *NAME* with your own name. Next create the batch file TREEPRT.BAT that contains the following.

```
@ECHO OFF
ECHO Turn your printer on line
ECHO Press CTRL-PrtSc to print a directory tree
PAUSE
TREE C:\
ECHO Press CTRL-PrtSc to disable printer echoing
```

Invoke the batch file TREEPRT, as shown here:

```
C:\BATCH> TREEPRT <ENTER>
```

When your screen displays the prompt for you to press CTRL-PRTSC, do so to enable printer echoing, as discussed in Chapter 13. DOS will in turn write the TREE command's output to your screen and printer. When the batch file completes, press CTRL-PRTSC again to disable printer echoing.

Execute the batch file TREEPRT again.

```
C:\BATCH> TREEPRT <ENTER>
```

This time, when PAUSE displays the message to press any key to continue, press CTRL-C to end the batch file. DOS will display the following prompt:

```
Terminate batch file (Y/N)?
```

Type Y and DOS will end the batch file, redisplaying the DOS prompt.

Edit the batch file TREEPRT.BAT and use the REM command to comment the batch file:

```
@ECHO OFF
REM Written by: Kris Jamsa
REM Date written: 2-26-93
REM Function: Lets the user print and display the directory tree.

REM Prompt the user to enable printer echoing. Pause until the
REM user presses a key to continue.
ECHO Turn your printer on line
ECHO Press CTRL-Prtsc to print a directory tree
PAUSE
```

```
REM Display the directory tree.
TREE C:\

REM Remind the user to disable printer echoing.
ECHO Press CTRL-PrtSc to disable printer echoing
```

Review

1. What is a batch file?
2. How does ECHO OFF differ from @ECHO OFF?
3. List two uses of the ECHO command.
4. Create a batch file named JUNE93.BAT that displays the calendar for June 1993, as shown here:

```
S    M    T    W    T    F    S
          1    2    3    4    5
6    7    8    9    10   11   12
13   14   15   16   17   18   19
20   21   22   23   24   25   26
27   28   29   30
```

5. What is the purpose of the PAUSE command?
6. How does ECHO OFF affect PAUSE?

Answers

1. A batch file is a user-defined file with the BAT extension that contains the names of one or more DOS commands. You execute a batch file by typing its name at the DOS prompt. When DOS encounters the batch file, it will automatically execute each of the commands the batch file contains.
2. If you are using DOS 3.3 or later, you can precede a batch file command with the @ character ("at" sign) to suppress the display of the command's name when the batch file executes. If a batch file contains the ECHO OFF command, the batch file will display the following on your screen:

```
C:\BATCH> ECHO OFF
```

If the batch file uses @ECHO OFF instead, the command name and prompt is not displayed.

3. The ECHO command lets your batch files disable (ECHO OFF) and enable (ECHO ON) command-name display. In addition, ECHO lets your batch files display messages to the user.
4. The batch file named JUNE93.BAT that displays the calendar for June 1993 is

```
@ECHO OFF
ECHO      S   M   T   W   T   F   S
ECHO              1   2   3   4   5
ECHO      6   7   8   9   10  11  12
ECHO      13  14  15  16  17  18  19
ECHO      20  21  22  23  24  25  26
ECHO      27  28  29  30
```

5. The PAUSE command lets your batch file display a message to the user and temporarily suspend the batch file's execution until the user presses any key to continue. Most batch files use the PAUSE command to suspend processing when they need the user to perform manual processing, such as turning on the printer or inserting a disk.
6. The PAUSE command displays an optional user message followed by a message for the user to press any key to continue. If the batch file uses the ECHO OFF command, PAUSE's optional user message does not appear, and the user only sees the prompt to press any key to continue. As a solution, most batch files use ECHO to display PAUSE's optional message and PAUSE to suspend processing, as shown here:

```
@ECHO OFF
ECHO Place a disk in drive A
PAUSE
TREE A:
```

Advanced Concepts

If you are using DOS 5 or later, the DOSKEY command discussed in Chapter 27 lets you create single-line macros that behave similarly to DOS batch files. DOSKEY macros, however, reside in RAM, which means DOS can execute the macros faster than it can execute your batch files. For more information on DOSKEY macros, turn to Chapter 27.

Using ECHO to Display Blank Lines

The ECHO command lets you disable command-name display and also write messages to the screen. As the complexity of your batch files increases, there may be times you want to improve your screen appearance by displaying blank lines. Unfortunately, using ECHO to display blank lines is harder than simply placing ECHO commands in a batch file, as shown here:

```
@ECHO OFF
ECHO Skip one line
ECHO
ECHO Skip two lines
ECHO
ECHO
ECHO DONE
```

By default, if you invoke ECHO without specifying ON, OFF, or a message, ECHO displays its current state of command-name display (either on or off).

```
C:\> ECHO <ENTER>
ECHO is on
```

If you invoke the previous batch file, your screen will display the following.

```
C:\BATCH> SKIPLINE <ENTER>
Skip one line
ECHO is off
Skip two lines
ECHO is off
ECHO is off
DONE
```

Rather than displaying blank lines as desired, the ECHO commands displayed ECHO's current status.

Depending on your DOS version, there are several techniques you can use to echo blank lines. First, the most commonly used method for echoing a blank line is to use a special extended ASCII blank character. As discussed in Chapter 10, you create this special blank character by holding down the ALT key and typing **255** on your keyboard's *numeric* keypad. When you release the ALT key, DOS generates the blank character. You cannot generate this character using the SPACEBAR. Most batch files that appear in books or magazines represent this invisible character as ALT-255. For example, the previous batch file would appear as follows:

```
@ECHO OFF
ECHO Skip one line
ECHO ALT-255
ECHO Skip two lines
ECHO ALT-255
ECHO ALT-255
ECHO DONE
```

In addition to using ALT-255, many DOS versions let you place any one of the following characters immediately next to ECHO to produce a blank line:

```
[ ] + / \ ; , .
```

Using a period to produce blank lines, the previous batch file becomes the following:

```
@ECHO OFF
ECHO Skip one line
ECHO.
ECHO Skip two lines
ECHO.
ECHO.
ECHO DONE
```

Experiment with these characters using ECHO under your version of DOS.

Using Extended ASCII Characters

Appendix B lists all the extended ASCII characters, such as the ALT-255 special blank character. Using these extended ASCII characters, you can improve your batch file's screen output by boxing menus or messages, as previously shown.

Creating the box-drawing characters requires the same steps as creating the ALT-255 space character. Simply hold down the ALT key and type the extended ASCII value. Figure 15-1 illustrates the box-drawing characters and their corresponding values.

Using these characters, you can create BOX.BAT, as shown here:

```
@ECHO OFF
ECHO ┌───┐
ECHO │BOX│
ECHO └───┘
```

Using ^G (CTRL-G) to Create Sound

PAUSE temporarily suspends a batch file's processing until you press a key. Unfortunately, if you have your back to the computer, the batch file may suspend for some time before you notice the prompt to press a key to continue. When you want the batch file to get the user's attention, you can echo the ^G character, which directs DOS to sound your computer's built-in bell. You can create the ^G character by holding down CTRL and pressing G. The following command creates a batch file named BELLS.BAT, which sounds the computer's speaker three times:

```
C:\BATCH> COPY CON BELLS.BAT <ENTER>
@ECHO OFF <ENTER>
ECHO ^G^G^G <ENTER>
^Z <ENTER>
        1 File(s) copied
```

By using your computer's bell in this way, your batch files can quickly notify the user of errors or other important events.

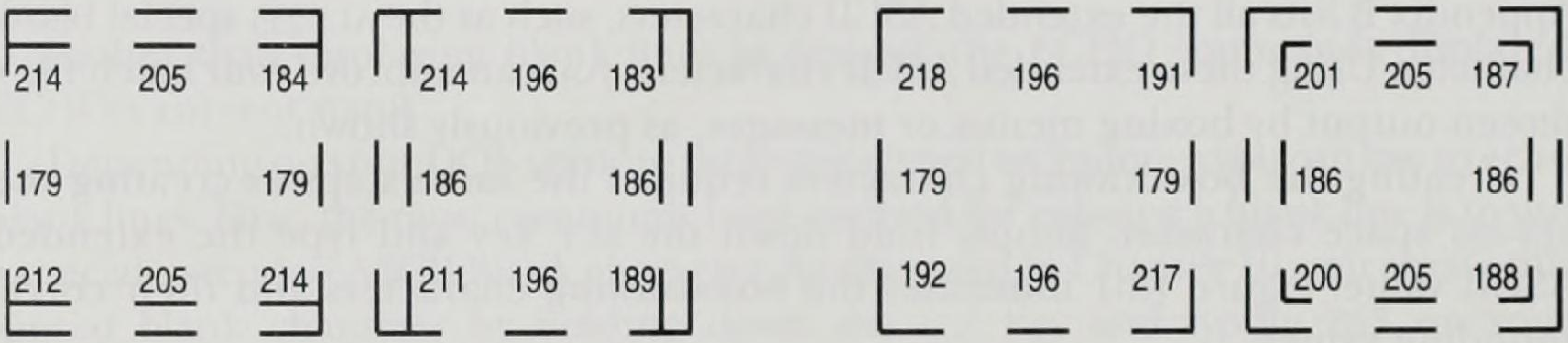

214
205
184
214
196
183
218
196
191
201
205
187
179
179
186
186
179
179
186
186
212
205
214
211
196
189
192
196
217
200
205
188

Figure 15-1. *Extended ASCII box characters and values*

Key Terms

Batch A group of one or more. In the case of DOS batch files, a batch is one or more related commands.

Batch file A user-defined file with the BAT extension that contains the names of one or more DOS commands. When you type a batch file's name at the DOS prompt, DOS executes, in order, each of the commands the batch file contains.

Interactive processing The process of executing a command, waiting for the command to complete, and then typing the next command to repeat the cycle.

Chapter 16

Decision-Making Batch Files

In Chapter 15, you learned that DOS batch files can save you time and keystrokes by letting you place one or more long or difficult commands into a file with the BAT extension. Like DOS commands, you execute your batch files by typing the batch file's name at the DOS prompt. When DOS encounters the batch file, DOS automatically executes all the commands the batch file contains. In Chapter 15, DOS executed batch file commands, in order, beginning with the first command in the batch file and ending with the last. Depending on your application, there may be times you'll want the batch file to execute one set of commands and not another. For example, assume you are writing a batch file that helps a new user get started with WordPerfect. If the user invokes the batch file from a directory other than WP, the batch file may need to perform additional CHDIR commands, or it may want to display a message to the user explaining how to use CHDIR to select the WP directory before starting WordPerfect. To perform such "custom" processing, your batch files must be able to make decisions on their own.

In this chapter you'll learn how to build decision-making batch files using the IF command. In addition, you'll learn how to expand your batch file's capabilities by letting the user provide information to the batch file, such as a desired filename or directory. Chapter 15 illustrated the convenience of batch files. This chapter introduces their true power.

The IF Command

The IF command provides your batch files with decision-making capabilities. In general, the IF command lets your batch files ask DOS if a specific condition is met. If so, the IF command executes a corresponding DOS command. The general format of the IF command is

IF *CONDITION DOS_COMMAND*

The IF command supports three different conditions. The first condition lets your batch file test whether a specific file exists; the second determines whether the previous batch file command was successful; the third condition tests whether two words or phrases are the same. Table 16-1 briefly presents the IF command's three condition types.

Regardless of the condition type, the IF command performs the same processing. If the condition is true, IF executes the command that follows. If the condition is not true, IF does not execute the command, and the batch file continues its processing with the next command.

IF EXIST Tests for Files

As your batch files become larger and more complex, they should behave intelligently. For example, a batch file should not use TYPE to display a file that doesn't exist. Likewise, a batch file that copies files from one directory to another should not overwrite existing files. Using the IF EXIST command, your batch files can find out if a specific file exists and act accordingly. For example, the following batch file

Condition	Purpose	Example
IF EXIST	Test if a file exists on disk	IF EXIST AUTOEXEC.BAT TYPE AUTOEXEC.BAT
IF ERRORLEVEL	Tests the success level of the previous batch command	IF ERRORLEVEL 1 GOTO SUCCESS
IF *STR*==*STR*	Tests if two character strings are the same	IF '%1'=='A' GOTO ASCENDING

Table 16-1. The IF Command's Three Condition Types

IMPTROOT.BAT uses the TYPE command to display the contents of AUTOEXEC.BAT and CONFIG.SYS, which reside in your startup disk's root directory. Each time you start DOS, the DOS startup software searches the root for these two important files and uses their contents to customize itself in memory for your particular configuration.

```
TYPE  \AUTOEXEC.BAT
TYPE  \CONFIG.SYS
```

The problem with this batch file is that it assumes the files exist. If the files are present, the batch file successfully displays their contents. However, if one or both of the files don't exist, the TYPE command will fail, displaying the "File not found" error message.

```
C:\BATCH> IMPTROOT <ENTER>

C:\BATCH> TYPE  \AUTOEXEC.BAT
File not found

C:\BATCH> TYPE  \CONFIG.SYS
File not found
```

To correct this possible problem, include the IF EXIST command:

```
IF EXIST  \AUTOEXEC.BAT  TYPE  \AUTOEXEC.BAT
IF EXIST  \CONFIG.SYS  TYPE  \CONFIG.SYS
```

As before, if either file exists, its contents are displayed. However, if a file does not exist, the IF command will not execute the corresponding TYPE command. Instead, the batch file's execution continues with the next command. Figure 16-1 uses a flowchart to illustrate the process.

To understand the flowchart, simply follow the direction of the arrows out of each condition box. If the condition is true, follow the arrow labeled "Yes." Likewise, if the condition is not true, follow the arrow labeled "No."

The NOT Operator

Just as there are times when you want IF to perform a command when a condition is true, so there may also be times you want IF to perform a command when a condition is false. For example, the IF EXIST command lets your batch file determine whether a file exists, and if so, to process accordingly. Many batch files, however, may want to perform specific processing when a file does not exist. In such cases, your batch files can use the logical operator, NOT.

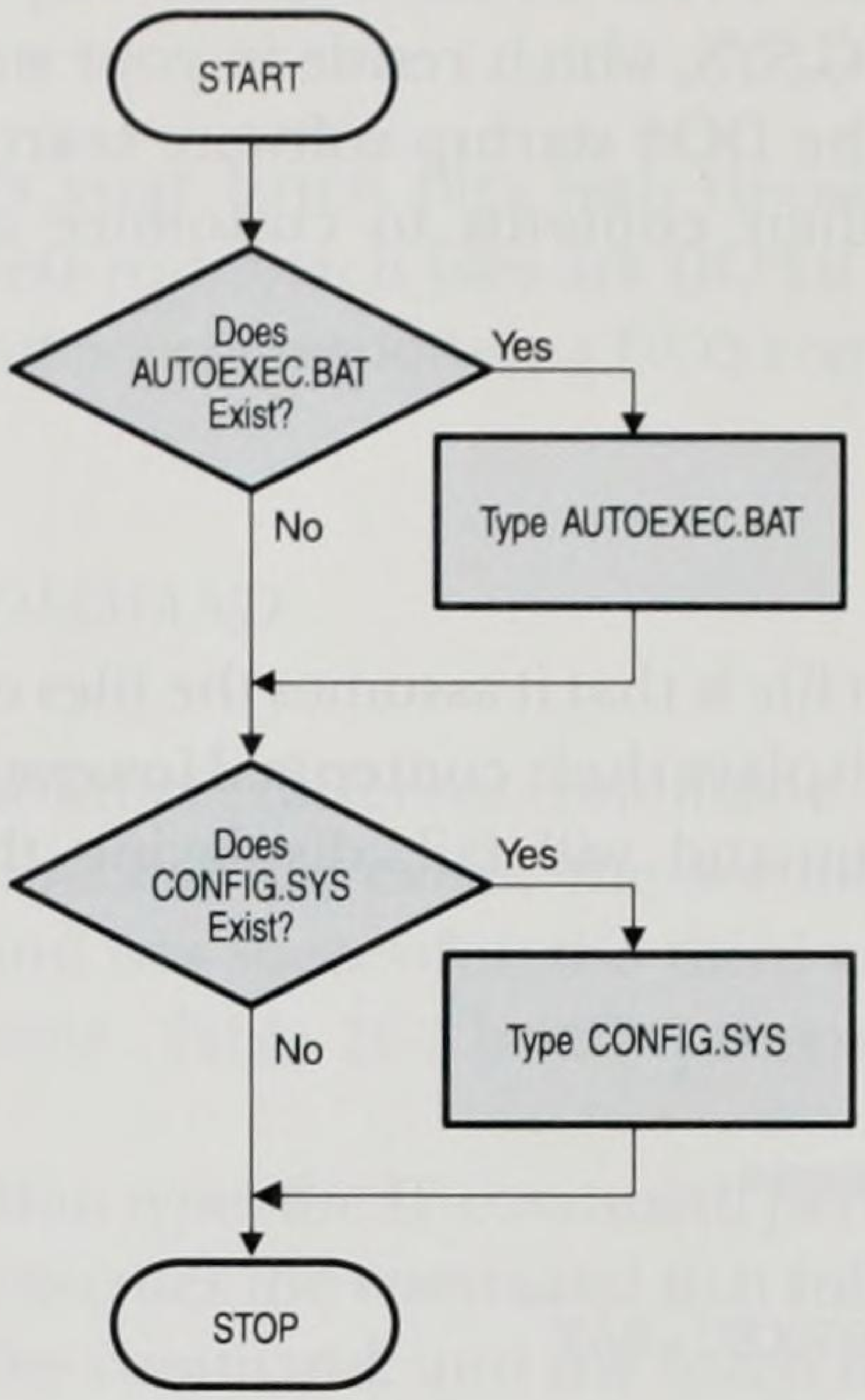


Figure 16-1. Charting IMPTROOT.BAT's possible execution paths

The NOT operator follows the word "IF" and precedes the condition:

IF NOT *CONDITION DOS_COMMAND*

In general, NOT directs IF to perform the command specified when the condition is false (NOT true). In the case of IF EXIST, the NOT operator directs IF to execute the command if the file specified does not exist.

The following batch file, CHECKFOR.BAT, searches the root directory for three files, AUTOEXEC.BAT, CONFIG.SYS, and NOMATCH.EXT. Using IF EXIST and IF NOT EXIST, the batch file displays a message that states whether or not each file is present.

```
@ECHO OFF
IF EXIST \AUTOEXEC.BAT  ECHO Found AUTOEXEC.BAT
IF EXIST \CONFIG.SYS ECHO Found CONFIG.SYS
IF EXIST \NOMATCH.EXT ECHO Found NOMATCH.EXT
IF NOT EXIST \AUTOEXEC.BAT  ECHO No such file: AUTOEXEC.BAT
IF NOT EXIST \CONFIG.SYS ECHO No such file: CONFIG.SYS
IF NOT EXIST \NOMATCH.EXT ECHO No such file: NOMATCH.EXT
```

The batch file's first three IF commands test whether the file exists. The last three IF commands use the NOT operator to test whether the file does not exist. Assuming your root directory has the files AUTOEXEC.BAT and CONFIG.SYS but not NOMATCH.EXT, your screen will display the following when you execute CHECKFOR.BAT:

```
C:\BATCH> CHECKFOR <ENTER>
Found AUTOEXEC.BAT
Found CONFIG.SYS
No such file: NOMATCH.EXT
```

If you want your batch file to perform one command when a condition is true and another command when the condition is false, your batch file will need two IF commands, one of which uses the NOT operator.

A batch file that copies one file to a different disk or directory should not overwrite an existing file. For example, the following batch file CHECK1ST.BAT copies the files AUTOEXEC.BAT and CONFIG.SYS to a disk in drive A. The batch file only copies the files if they don't exist on drive A. If a file exists, the batch file displays a message stating that the disk already contains the file.

```
@ECHO OFF
IF EXIST A:AUTOEXEC.BAT ECHO AUTOEXEC.BAT exists on drive A
IF NOT EXIST A:AUTOEXEC.BAT COPY \AUTOEXEC.BAT A:
IF EXIST A:CONFIG.SYS ECHO CONFIG.SYS exists on drive A
IF NOT EXIST A:CONFIG.SYS COPY \CONFIG.SYS A:
```

Figure 16-2 uses a flowchart to illustrate the batch file's decision making.

All the examples presented so far, only execute one command when a condition is either true or false. In Chapter 17, you will learn how to use the GOTO command to execute several commands for one condition and several other commands for a different condition.

IF ERRORLEVEL Tests for Success

As your batch files increase in size, there will be times when you only want the batch file's execution to continue if the previous command was successful. In addition to displaying error messages on your screen, many DOS commands provide DOS with an exit status value that indicates whether or not the command was successful. Using the IF ERRORLEVEL command, your batch files can test this status value and process accordingly.

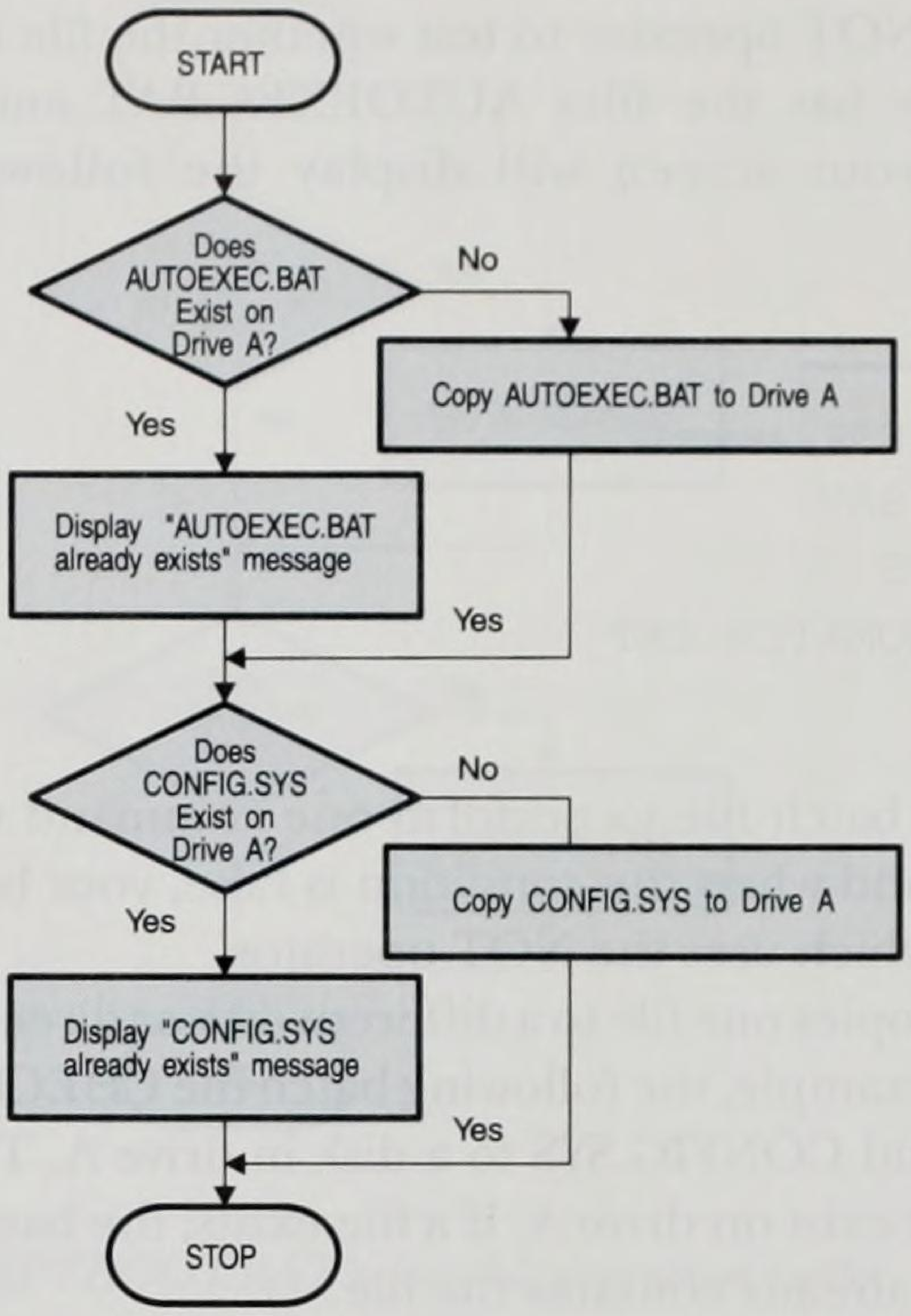

START
Does AUTOEXEC.BAT Exist on Drive A?
No
Copy AUTOEXEC.BAT to Drive A
Yes
Display "AUTOEXEC.BAT already exists" message
Yes
Does CONFIG.SYS Exist on Drive A?
No
Copy CONFIG.SYS to Drive A
Yes
Display "CONFIG.SYS already exists" message
Yes
STOP

Figure 16-2. *Charting CHECK1ST.BAT's execution paths*

For example, in Chapter 12, you learned that DISKCOPY copies the contents of one floppy disk to another. When DISKCOPY completes, DISKCOPY returns one of the status values listed in Table 16-2 to DOS.

If, for example, you have a batch file that uses DISKCOPY, you can place IF ERRORLEVEL commands immediately after the DISKCOPY command to determine whether DISKCOPY was successful and then process accordingly.

Status Value	Meaning
0	Successful disk copy operation
1	Recoverable disk read or write error
2	CTRL-C termination
3	Fatal disk read or format error
4	Invalid disk drive, syntax, or insufficient memory error

Table 16-2. *Exit Status Values Produced by DISKCOPY*

IF ERRORLEVEL lets your batch files test for a specific exit status value. Like IF EXIST, the IF ERRORLEVEL command has a condition and a corresponding command DOS executes when the condition is true. For IF ERRORLEVEL, a condition is true when the previous command's exit status value is *greater than or equal to* the value specified. For example, the following IF ERRORLEVEL command tests for an exit status value *greater than or equal to* 3.

```
IF ERRORLEVEL 3 DOS_COMMAND
```

In this case, if the previous command exits with a status value of 3 or above, DOS will execute *DOS_COMMAND.* Because IF ERRORLEVEL also considers exit status values *greater* than the value specified as successful, you need to take care in creating your batch files.

The following batch file, WRONGIF.BAT uses IF ERRORLEVEL incorrectly to determine whether DISKCOPY was successful:

```
@ECHO OFF
DISKCOPY A:  B:
IF ERRORLEVEL 0 ECHO Successful disk copy!
```

As you can see in Table 16-2, DISKCOPY returns the exit status 0 to DOS when it successfully copies a disk. Unfortunately, the batch file uses IF ERRORLEVEL incorrectly. Remember, IF ERRORLEVEL evaluates to true for any exit status value that is greater than or equal to the value specified—in this case, 0. In DISKCOPY, every possibility is greater than or equal to 0. No matter how DISKCOPY completes, the batch file will display the "Successful disk copy!" message. Verify this for yourself by creating and invoking the batch file. When DISKCOPY prompts you to press any key to continue, use the CTRL-C key combination to end the command. Although the batch file did not perform the disk copy operation, batch file displays a success message:

```
C:\BATCH> CHKDCOPY <ENTER>
Insert SOURCE diskette in drive A:

Insert TARGET diskette in drive B:

Press any key to continue . . .
^C
Successful disk copy!
```

The best way to prevent this type of confusion is to combine two IF ERRORLEVEL commands. The first command tests whether the exit status value is greater than or equal to the desired value. The second command uses the NOT operator to test whether the exit status is NOT greater than the value. By combining the two IF commands, your batch file can test for an exact value. For example, to test for an exit status value of 0, your batch file would use the following:

```
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 DOS_COMMAND
```

Likewise, to test for an exit status of 1, your batch file would use the following:

```
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DOS_COMMAND
```

Using two IF ERRORLEVEL commands each to test for exact values, the following batch file CHKDCOPY.BAT displays the correct message for each of DISKCOPY's possible exit status values:

```
@ECHO OFF
DISKCOPY A: B:
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 ECHO Successful disk copy!
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO Recoverable disk error
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO User Ctrl-C termination
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO Fatal disk read or format
IF ERRORLEVEL 4 ECHO Invalid disk drive, syntax, or memory error
```

If you cancel CHKDCOPY.BAT using CTRL-C, you will display the following:

```
C:\BATCH> CHKDCOPY <ENTER>
Insert SOURCE diskette in drive A:

Insert TARGET diskette in drive B:

Press any key to continue . . .
^C
User Ctrl-C termination
```

Unfortunately, not all DOS commands provide exit status values your batch files can test. Appendix A lists the exit status values provided by DOS commands.

In Chapter 29 you will learn how to create your own simple commands using the DEBUG command. Many of these commands will return exit status values your batch files can use. One such command, HOUR.COM, returns the current hour of the day. Using IF ERRORLEVEL and HOUR.COM, the following batch file, GREET.BAT, displays either the message "Good morning!", "Good afternoon!", or "Good evening!", depending on the current time.

```
@ECHO OFF
HOUR
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 12 ECHO Good morning!
IF ERRORLEVEL 12 IF NOT ERRORLEVEL 18 ECHO Good afternoon!
IF ERRORLEVEL 18 ECHO Good evening!
```

As you work more with DOS, you will encounter many more simple commands like HOUR.COM you can put to work in your batch files.

Using IF to Compare Two Character Strings

A *character string* is a sequence of characters. The sequence might be a single letter, a word, a phrase, or zero letters (the null string). The third form of the IF command lets your batch files compare two strings and execute a command if the strings are equal. The IF command specifies the strings to compare by separating the strings with two equal signs.

```
IF String1==String2 DOS_COMMAND
```

NOTE

The IF command only considers two strings as the same if the strings match letter for letter, including matching either upper- or lowercase. IF would not, therefore, consider the strings "ABC" and "abc" as the same. The best way to understand this form of the IF command is to use it with batch parameters, discussed next.

Passing Information to Your Batch Files

As you work more with DOS batch files, you will find that a batch file you created to simplify one task could simplify several others if you could only change the name of a file the batch file uses. For such cases, DOS lets you pass information to your batch files using *batch parameters.*

So far, each time you have executed a batch file, you have simply typed the batch filename. If you include additional information on the command, such as a filename or drive letter, DOS makes that information available to your batch file. A batch parameter, then, is simply a value you specify in the batch command line.

To access the parameters within your batch files, you use the symbols %1 through %9. DOS assigns the first parameter to %1, the second to %2, and so on. If you don't specify values for a parameter, DOS assigns the parameter the *null string* or empty string. Using the IF command, your files can test for a null string. DOS assigns the batch filename to the parameter %0.

The following batch file, SHOWALL.BAT, uses ECHO to display the values of the parameters %0 through %9.

```
@ECHO OFF
ECHO This batch file is %0
ECHO The parameters are %1 %2 %3 %4 %5 %6 %7 %8 %9
```

If you invoke SHOWALL with no parameters, your screen will display the following:

```
C:\BATCH> SHOWALL <ENTER>
This batch file is SHOWALL
The parameters are
```

In this case, DOS assigns the null string to the parameters %1 through %9. If you instead invoke the batch file with the letters A, B, and C, your screen will display the following:

```
C:\BATCH> SHOWALL  A  B  C <ENTER>
This batch file is SHOWALL
The parameters are A B C
```

Figure 16-3 illustrates how DOS assigns the parameter values.

If you invoke the batch file with the letters A through L, your screen will display the following:

```
C:\BATCH> SHOWALL  A  B  C  D  E  F  G  H  I  J  K  L <ENTER>
This batch file is SHOWALL
The parameters are A B C D E F G H I
```

You can see that DOS has assigned values to the parameters %1 through %9. In Chapter 17 you will learn how to access the values not assigned to a parameter with the SHIFT command.

Now that you know how batch parameters work, you need to put the parameters to work. The following batch file, COLOR.BAT, uses the IF command to compare the value in %1 to several color names. If the batch file locates a match, it displays a message on your screen.

```
@ECHO OFF
IF '%1'=='BLUE' ECHO The color is BLUE
IF '%1'=='RED' ECHO The color is RED
IF '%1'=='GREEN' ECHO The color is GREEN
IF '%1'=='' ECHO No color specified
```

Remember, the IF command only considers two strings the same if the letters match in case. If you invoke COLOR.BAT with "Blue", the batch file will not find a match.

```
C:\BATCH> COLOR Blue
C:\BATCH>
```

After DOS assigns %1 the value Blue, the batch file actually performs the following comparisons.

```
@ECHO OFF
IF 'Blue'=='BLUE' ECHO The color is BLUE
IF 'Blue'=='RED' ECHO The color is RED
IF 'Blue'=='GREEN' ECHO The color is GREEN
IF 'Blue'=='' ECHO No color specified
```

Take a close look at the following IF command:

```
IF '%1'==''  ECHO No color specified
```

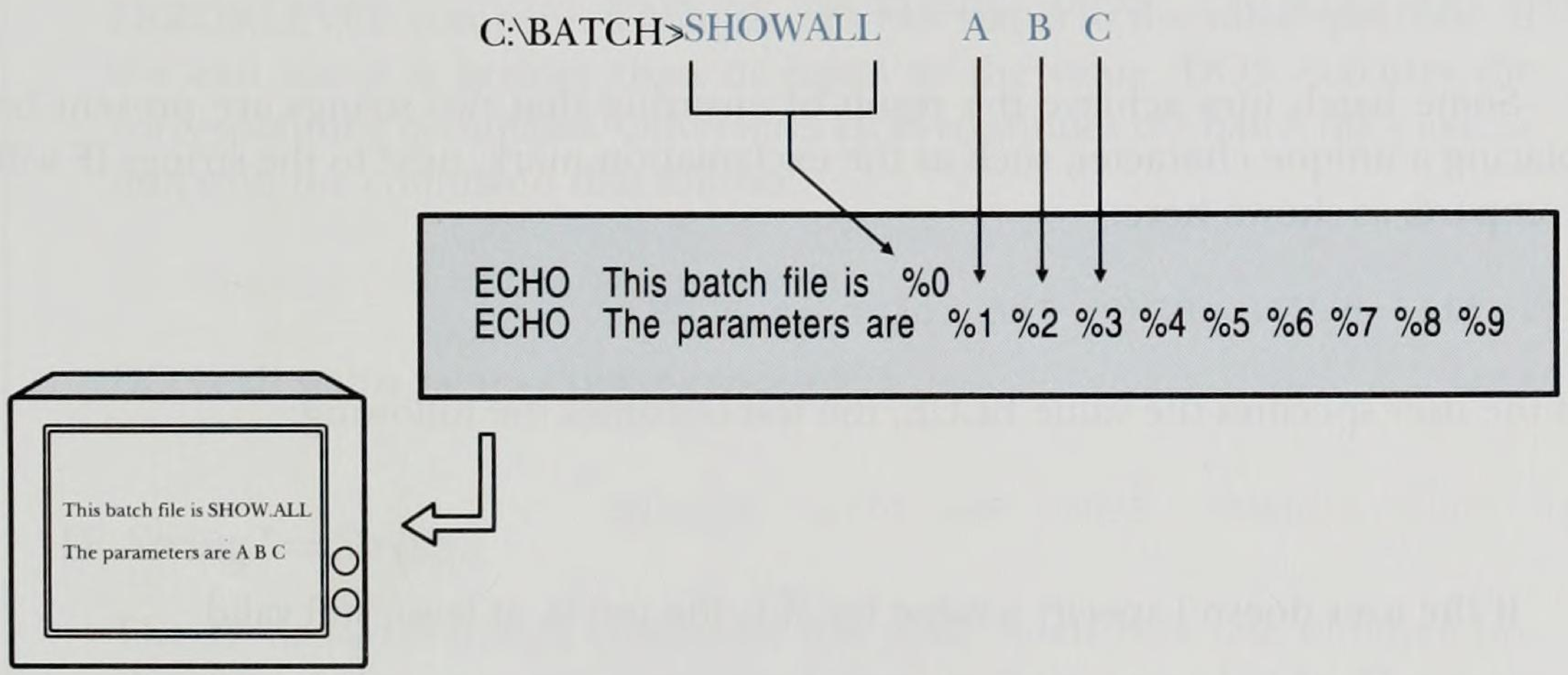

C:\BATCH>SHOWALL A B C
ECHO This batch file is %0
ECHO The parameters are %1 %2 %3 %4 %5 %6 %7 %8 %9
This batch file is SHOW.ALL
The parameters are A B C

Figure 16-3. *Assigning A, B, and C to %1, %2, and %3*

If you don't specify a parameter's value, DOS assigns the null string to the parameter. The two single quotes *with no space between* represent the null string. If %1 has no value, the IF command becomes the following:

```
IF ''==''  ECHO No color specified
```

Because the two null strings are the same, DOS executes the ECHO command.

When you compare strings using the IF command, you must make sure a string appears on both sides of the double equal signs. Assume, for example, you used the following IF command:

```
IF  %1==BLUE  ECHO  The color is BLUE
```

If the user does not specify a value for %1, the IF command becomes

```
IF  ==BLUE  ECHO  The color is BLUE
```

Because only one side of the double equal signs contains a string, DOS will display the following message informing you that the command violates the syntax or format of the IF command:

```
IF  ==BLUE  ECHO  The color is BLUE
Syntax error
```

To ensure that each side always has at least a null string, always surround the strings with single quotes, as shown here:

```
IF  '%1'=='BLUE'  ECHO  The color is BLUE
```

Some batch files achieve the result of ensuring that two strings are present by placing a unique character, such as the exclamation mark, next to the strings IF will compare, as shown here:

```
IF  %1!==BLUE!  ECHO  The color is BLUE
```

If the user specifies the value BLUE, the test becomes the following:

```
IF  BLUE!==BLUE!  ECHO  The color is BLUE
```

If the user doesn't specify a value for %1, the test is, at least, still valid.

```
IF  !==BLUE!  ECHO  The color is BLUE
```

Although using a unique character in this way is valid, it is harder to understand than the pair of single quotes. At first glance, many users might think they needed to specify the color with an exclamation mark, such as "BLUE!", with perhaps puzzling results (BLUE!!= =BLUE!).

If you create a batch file that uses batch parameters, use the IF command to ensure that the batch parameters are defined. In Chapter 17, you will create several powerful batch files that use batch parameters to specify filenames and DOS wildcards. For now, simply recall that DOS assigns batch parameters to %1 through %9.

SUMMARY

IF EXIST

The IF EXIST command lets your batch files test whether a file exists, then process accordingly. The IF EXIST command specifies a file DOS is to search for and a command DOS executes if the file exists.

```
IF EXIST FILENAME.EXT DOS_COMMAND
```

Using the NOT operator, your batch file can execute a different command if the file does not exist.

```
IF NOT EXIST FILENAME.EXT DOS_COMMAND
```

IF ERRORLEVEL

The IF ERRORLEVEL command lets your batch files determine the success of the previous batch file command and process accordingly. IF ERRORLEVEL compares a command's exit status to the value specified. If the exit status is greater than or equal to the value, DOS executes the corresponding command. Otherwise, DOS continues the batch file's execution with the command that follows:

```
IF ERRORLEVEL value DOS_COMMAND
```

DOS exit status values cannot exceed 255.

IF *String1==String2*

The IF *String1==String2* command lets your batch files test whether two character strings are the same, and then process accordingly. The most common use of this form of the IF command is for testing batch parameters. For DOS to consider two strings as the same, the strings must match letter

SUMMARY

for letter. DOS does not consider an uppercase letter the same as its lowercase counterpart.

```
IF String1==String2 DOS_COMMAND
```

Batch Parameters

As DOS lets you pass filenames and disk drive letters to your DOS commands, so the same is true for batch files. When you include additional information in your batch command lines, DOS assigns the first nine pieces of information to the batch parameters %1 through %9. Using these parameters, a batch file becomes more general and can support more than one application. In addition to %1 through %9, DOS assigns the batch filename to %0.

Hands On

If the BATCH directory is not the current directory, use CHDIR to select it. Next, create the following batch file TESTAUTO.BAT that tests if the file AUTOEXEC.BAT is in the root directory. If the file exists, the batch file displays the file's contents; otherwise, the batch file displays the message "\AUTOEXEC.BAT does not exist."

```
@ECHO OFF
IF EXIST \AUTOEXEC.BAT TYPE \AUTOEXEC.BAT
IF NOT EXIST \AUTOEXEC.BAT ECHO \AUTOEXEC.BAT does not exist
```

Next, create the batch file TESTFILE.BAT that uses the %1 batch parameter instead of AUTOEXEC.BAT to let the batch file support any filename.

```
@ECHO OFF
IF EXIST %1 TYPE %1
IF NOT EXIST %1 ECHO %1 does not exist
```

Invoke TESTFILE.BAT with the filename AUTOEXEC.BAT:

```
C:\BATCH> TESTFILE \AUTOEXEC.BAT <ENTER>
```

When you use the batch parameters %1 through %9, you need to use the IF command to test whether the parameter is defined. To understand why, invoke TESTFILE without a filename.

```
C:\BATCH> TESTFILE <ENTER>
Bad command or file name
```

The test case will result in this error message because the first command line becomes

```
IF EXIST TYPE
```

which is a legal IF command that searches for a file named TYPE. Note, however, that the IF command does not provide a corresponding action if such a file exists. The second command line becomes

```
IF NOT EXIST ECHO DOES NOT EXIST
```

and DOS sees the word "DOES" as the bad command. Edit TESTFILE.BAT adding the following IF commands to test whether %1 is defined.

```
@ECHO OFF
IF NOT '%1'=='' IF EXIST %1 TYPE %1
IF NOT '%1'=='' IF NOT EXIST %1 ECHO %1 does not exist
```

Next create the batch file LISTPAR.BAT that displays each batch parameter, as shown here:

```
@ECHO OFF
ECHO %1
ECHO %2
ECHO %3
ECHO %4
ECHO %5
ECHO %6
ECHO %7
ECHO %8
ECHO %9
```

Experiment with LISTPAR.BAT, invoking it with 0, 3, 9, and 15 parameters.

Review

1. List three tests the IF command performs.
2. What is the purpose of the NOT operator?
3. How do the following IF commands differ?

```
IF ERRORLEVEL 3 DOS_COMMAND
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 DOS_COMMAND
```

4. What is a batch parameter?
5. Create a batch file, DEFINED.BAT, that examines the batch parameters %1 through %3. If a parameter is defined, display the parameter's value. If the parameter is not defined, display an appropriate error message.

Answers

1. The IF command lets you test whether a file exists on disk, the previous command was successful, and two character strings are the same.
2. The NOT operator directs DOS to perform the command specified in an IF command when the corresponding condition is false.
3. The first IF command executes *DOS_COMMAND* if the previous command's exit status value is greater than or equal to 3. The second IF command only executes *DOS_COMMAND* when the previous command's exit status value is exactly 3.
4. A batch parameter is a value, such as filename or disk-drive letter, that the user includes in the batch file's command line. DOS assigns the first nine batch parameters to the symbols %1 through %9 your batch files can use to access the parameter's value.
5. The batch file is

```
@ECHO OFF
IF '%1'=='' ECHO %%1 has No value
IF '%2'=='' ECHO %%2 has No value
IF '%3'=='' ECHO %%3 has No value
IF NOT '%1'=='' ECHO %%1 is %1
IF NOT '%2'=='' ECHO %%2 is %2
IF NOT '%3'=='' ECHO %%3 is %3
```

When DOS encounters the double percent signs, such as %%1 or %%2, DOS will ECHO the string %1 or %2.

Advanced Concepts

As you have learned, batch parameters increase your batch file flexibility. If you are using DOS version 3.3 or later, your batch files can use named parameters to access

DOS environment entries. DOS environment entries have a name and value. Using the SET command, you can display the current environment:

```
C:\BATCH> SET <ENTER>
PROMPT=$P$G
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS;C:\DISKUTIL;C:\BATCH
```

As you will see in Chapter 19, the environment provides a place for you to store information for DOS, or as you will see, your batch files. A batch file *named parameter* is simply an environment entry whose name appears between percent signs, such as %PATH%.

When DOS encounters a named parameter in your batch file, DOS substitutes the parameter's corresponding environment value. If the parameter is not defined in the environment, DOS replaces the parameter with the null string. The following batch file, SHOWENV.BAT, uses the named parameters %PATH% and %PROMPT% to display your current command path and system prompt:

```
@ECHO OFF
ECHO Current path is %PATH%
ECHO Current prompt is %PROMPT%
```

Given the previous environment values, your screen will display the following when you invoke SHOWENV.BAT:

```
C:\BATCH> SHOWENV <ENTER>
Current path is C:\DOS;C:\DISKUTIL;C:\BATCH
Current prompt is $P$G
```

As you can see, DOS substitutes the environment entry values as you desire. As you should always test batch parameters to ensure the parameters are defined, so should you test named parameters.

The following batch file modifies SHOWENV.BAT to test whether the named parameters %PATH%, %PROMPT%, and %BOOK% are defined:

```
@ECHO OFF
IF '%PATH%'=='' ECHO PATH is undefined
IF NOT '%PATH%'=='' ECHO PATH is %PATH%
IF '%PROMPT%'=='' ECHO PROMPT is undefined
IF NOT '%PROMPT%'=='' ECHO PROMPT is %PROMPT%
IF '%BOOK%'=='' ECHO BOOK is undefined
IF NOT '%BOOK%'=='' ECHO BOOK is %BOOK%
```

When you invoke this batch file, your screen displays the following:

```
C:\BATCH> SHOWENV <ENTER>
PATH is C:\DOS;C:\DISKUTIL;C:\BATCH
PROMPT is $P$G
BOOK is undefined
```

Use the SET command to assign a value to BOOK. Generally SET, discussed in Chapter 19, allows you to assign a value to a name. Let's assign BOOK the name READER, in the following example,

```
C:\BATCH> SET BOOK=READER
```

and repeat the SHOWENV command. The batch file will display the entry's new value as shown:

```
C:\BATCH> SHOWENV <ENTER>
PATH is C:\DOS;C:\DISKUTIL;C:\BATCH
PROMPT is $P$G
BOOK is READER
```

As the complexity of your batch files increases, you will use named parameters to save and restore such information as the current directory, current drive, or command path. Chapter 17 shows you how you can save and later restore the current directory using named parameters.

Creating Y_OR_N.COM

The true power of the IF ERRORLEVEL command isn't its use with DOS commands, but rather commands you create yourself using DEBUG (discussed in Chapter 29). For example, the following DEBUG script file Y_OR_N.SCR creates a command named Y_OR_N.COM your batch files can use to get a yes or no user response:

```
N Y_OR_N.COM
A 100
MOV AH, 0
INT 16
MOV AL, 0
CMP AH, 15
```

```
JE  112
CMP AH, 31
JE  114
JMP 100
MOV AL, 1
MOV AH, 4C
INT 21
<ENTER>
R CX
18
W
Q
```

Create Y_OR_N.COM by redirecting the file's contents into DEBUG, as shown in the following:

```
C:\BATCH> DEBUG  <  Y_OR_N.SCR <ENTER>
```

Y_OR_N.COM waits for the user to type either Y or N. If the user types Y, the command exits with the status value 1. If the user types N, Y_OR_N exits with 0. Using the IF ERRORLEVEL command, your batch files can determine the user's response, as shown here:

```
@ECHO OFF
ECHO Do you want to delete %1  (Y/N)?
Y_OR_N
IF ERRORLEVEL  1  DEL  %1
```

As you work with DOS batch files, you will collect a library of useful commands such as Y_OR_N.COM that increase your batch file flexibility.

Using the DOS 6 CHOICE Command

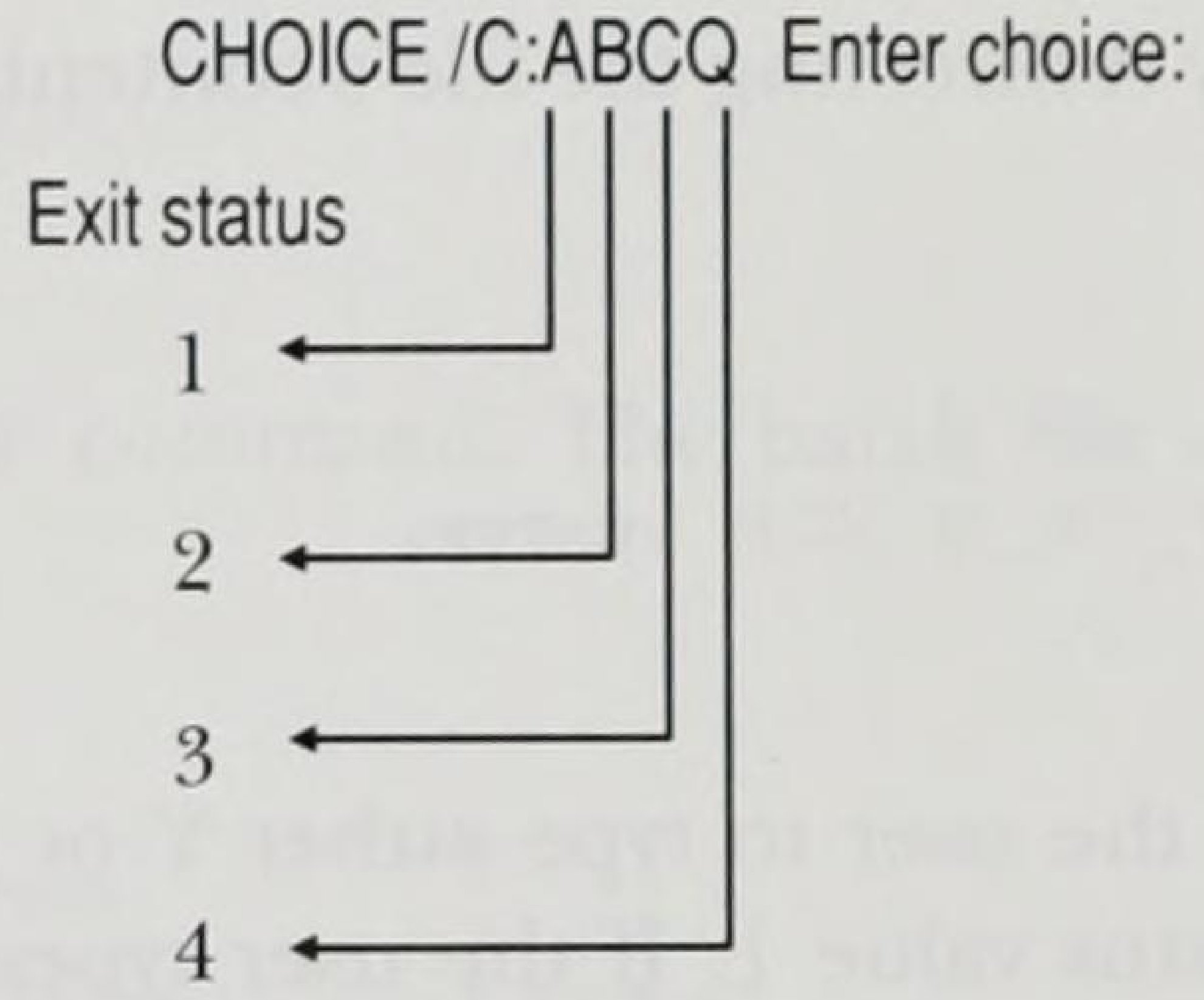
As you have learned, the DOS IF command lets your batch files make decisions based on different conditions. If you are creating a batch file for another user, you might want the batch file to display a menu of options similar to the following.

```
A    Display a directory listing
B    Invoke Lotus 1-2-3
C    Start WordPerfect
Q    Quit (to DOS)
Enter choice:[A,B,C,Q]?
```

When the user types **A**, **B**, **C**, or **Q**, the batch file will perform the desired operation. If you are using DOS 6, the CHOICE command lets your batch files wait for and determine a user response. The following CHOICE command waits for the user to type **A**, **B**, **C**, or **Q**:

```
CHOICE /C:ABCQ Enter choice:
```

The letters that follow the CHOICE /C switch specify the keys CHOICE will wait for. If the user types any other key, CHOICE will ignore the keystroke. When the user types one of the specified keys, CHOICE returns an exit status value that corresponds to the key pressed. Given the previous command, CHOICE will return the exit status 1 if the user types **A**, 2 for **B**, 3 for **C**, and 4 for **Q**, as shown here:



Should the user cancel the CHOICE command by pressing CTRL-C, CHOICE returns the exit status value 0. Within your batch file, you can use the IF ERRORLEVEL command to determine the key pressed. The following batch file, for example, displays a menu of options and then waits to determine the key pressed:

```
@ECHO OFF
:LOOP
CLS ECHO            Main Menu
ECHO.
ECHO    A       Display directory listing
ECHO    B       Display memory use
ECHO    C       Display CONFIG.SYS
ECHO    Q       Quit to DOS
ECHO.
CHOICE /C:ABCQ Enter choice:
IF NOT ERRORLEVEL 1 GOTO DONE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DIR /P
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 MEM /DEBUG /PAGE
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 MORE < \CONFIG.SYS
IF ERRORLEVEL 4 GOTO DONE
PAUSE
GOTO LOOP
:DONE
```

Specifying a Default CHOICE Option

In many cases, your menu-driven batch files will have one option that users will select most often. Therefore, CHOICE lets you specify a default option, which, if the user does not respond in a given number of seconds, CHOICE selects for them. Consider, for example, a batch file that asks the user a Yes or No question, such as whether or not they want to print the file CONFIG.SYS. The batch file could invoke the CHOICE command as follows:

```
CHOICE /C:YN Do you want to print CONFIG.SYS
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 PRINT \CONFIG.SYS
```

Assume that in most cases, the user selects the No option. Using CHOICE's /T switch, you can select N as the default. To do so, you must determine the number of seconds (from 0 to 99) after which, if the user fails to respond, CHOICE selects the default option. The following CHOICE command, for example, directs CHOICE to select the N option after 15 seconds, should the user fail to make a response:

```
CHOICE /C:YN /T:N,15 Do you want to print CONFIG.SYS
```

In other words, after 15 seconds, CHOICE behaves as if the user pressed the **N** key, in this case returning the exit status value 2.

In Chapter 18 you will use a special batch file named AUTOEXEC.BAT that executes each time your system starts. To control which commands AUTOEXEC.BAT executes, you might use the CHOICE command. By including default options and timeout periods, you can ensure that your system will start, even if you are away from the keyboard after you turn on the system's power.

The previous CHOICE command used the switch /C:YN to prompt the user for a Yes or No response. As it turns out, if you don't specify a /C switch, the default is /C:YN. As a result, the following commands are identical in function:

```
CHOICE /C:YN Do you want to print CONFIG.SYS
CHOICE Do you want to print CONFIG.SYS
```

Case-Sensitive Processing

By default, CHOICE considers upper- and lowercase letters the same. For example, the following CHOICE command waits for the user to type A, B, C, or Q:

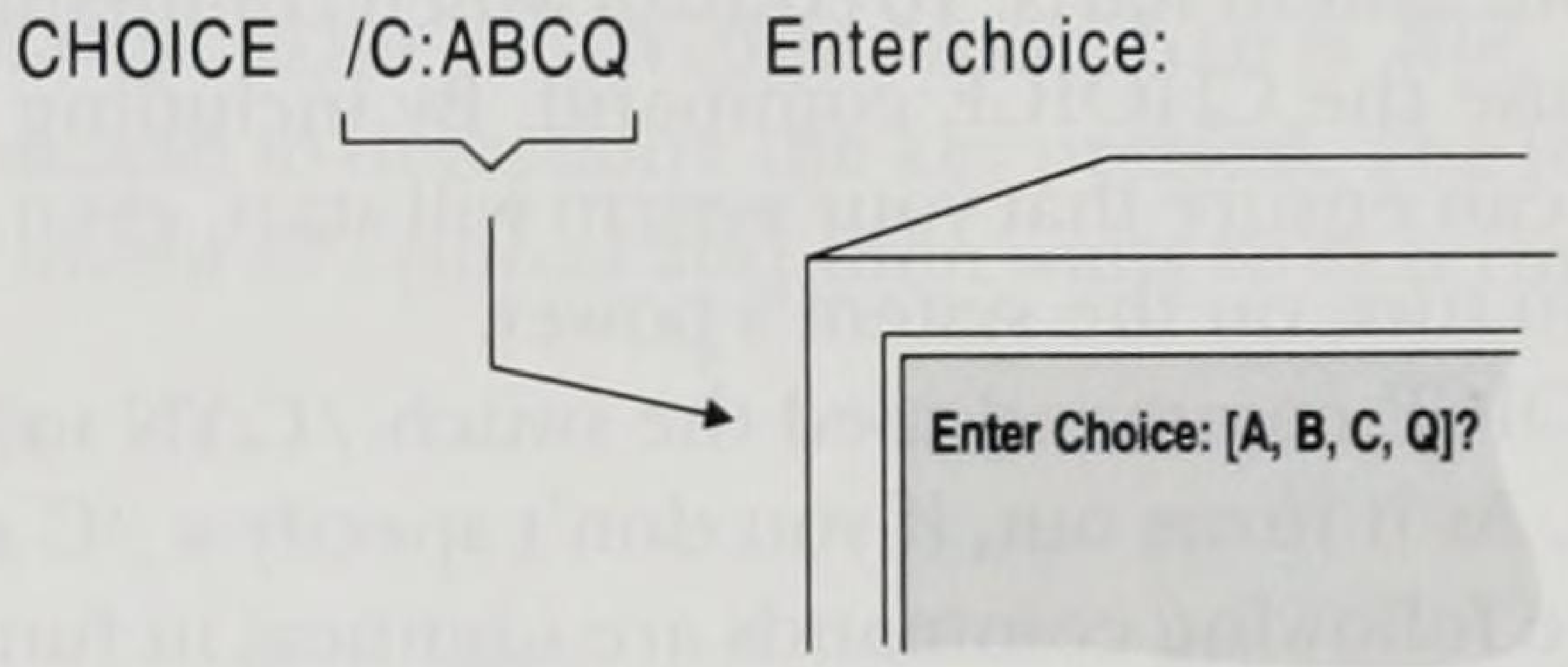
```
CHOICE /C:ABCQ Enter choice:
```

CHOICE will return the same exit status value if the user types A or a, or B or b. Depending on your batch file's processing, there may be times when you want CHOICE to distinquish between upper- and lowercase letters. To do so, simply include CHOICE's /S switch. The following batch file displays a menu that lets the user display a directory listing sorted in several different ways. The batch file supports upper- and lowercase letters.

```
@ECHO OFF
ECHO          Main Menu
ECHO.
ECHO     N    Sort by name A-Z
ECHO     n    Sort by name a-z
ECHO     S    Sort by size big to small
ECHO     s    Sort by size small to big
ECHO     E    Sort by extension A-Z
ECHO     e    Sort by extension a-z
ECHO.
CHOICE /C:NnSsEe /S Enter choice:
IF NOT ERRORLEVEL 1 GOTO DONE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DIR /O:N
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 DIR /O:-N
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 DIR /O:S
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 DIR /O:-S
IF ERRORLEVEL 5 IF NOT ERRORLEVEL 6 DIR /O:E
IF ERRORLEVEL 6 IF NOT ERRORLEVEL 7 DIR /O:-E
:DONE
```

Suppressing CHOICE's Display of Valid Keys

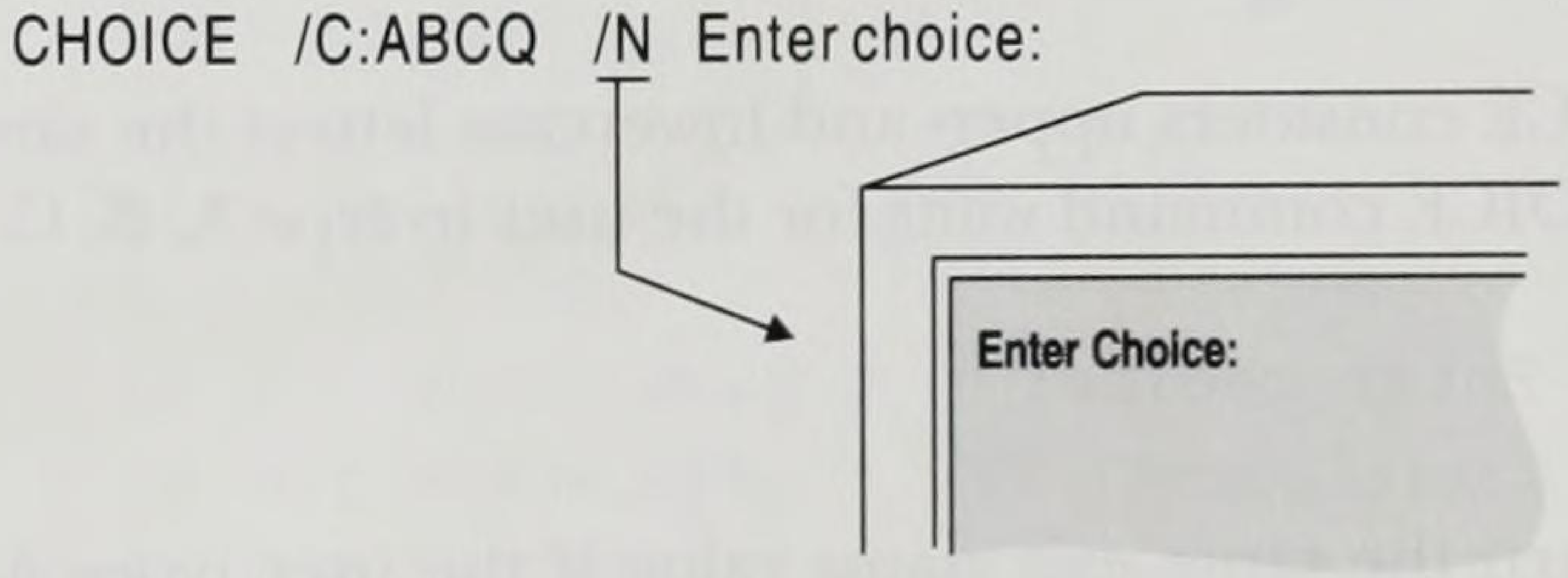
By default, CHOICE displays the keys the user can press in its prompt, as shown here:



```
CHOICE /C:ABCQ  Enter choice:
```

```
Enter Choice: [A, B, C, Q]?
```

If your batch file displays a menu with several different options, displaying the keys in the prompt is distracting. If you include the /N switch in the command line, CHOICE will not display the keys, as you can see here:



```
CHOICE /C:ABCQ /N Enter choice:
```

```
Enter Choice:
```

Key Terms

Condition A test producing a true or false result.

Exit status value A value from 0 to 255 that states a command's level of success.

Batch parameter Information that the user includes in the batch file's command line that the batch file can access using the symbols %1 through %9.

Named parameter In DOS 3.3 or later, an environment entry name surrounded by percent signs for which DOS substitutes the entry's value during the batch file's execution.

Chapter 17

Getting the Most from Batch Files

In Chapter 16 you learned that, using the IF command and batch parameters, DOS batch files can become very powerful. Chapter 1 defined a program as a list of instructions the computer is to perform. Although you may not have realized it, in Chapter 16 you became a batch file programmer. Not only did you provide DOS with a list of commands to perform, you used the IF command in many of your batch files so that they could make intelligent decisions about the processing they should perform. In this chapter you will round out your batch programming skills as you examine the remaining DOS batch commands.

Branching from One Location to Another Using GOTO

In Chapter 16 your batch files used the IF command to test for different conditions. When a condition was met, the batch file executed a specific command. In many cases, however, you will want your batch file to perform several related commands when a condition is met. To do so, you will group related commands, placing them at a specific location in your batch file. When the condition is met, your batch file will use the GOTO command to skip to the desired commands and then to the end of the batch file.

Until now, each batch has begun its execution at the first command at the top of the batch file, working its way through commands toward the bottom of the file. The GOTO command directs DOS to branch from one location in your batch file to another. The format of the GOTO command is GOTO *LABELNAME*.

A *label* is a name on a line in your batch file that serves as a marker or "landing site" for a GOTO jump. The first character of a label line is always a colon (:). This tells DOS that what follows on the line is a label name to be jumped to, not a command to be executed. The following batch file, SKIPDIR.BAT, uses GOTO to branch to the label DONE:

```
@ECHO OFF
VER
GOTO DONE
DIR
:DONE
```

When executed, this batch file displays the current DOS version number and then ends. The batch file never executes the DIR command because the GOTO command branches, or jumps, past the command, as shown in the following:

```
@ECHO OFF
VER
GOTO DONE
DIR
:DONE
```

When you specify a label name in a GOTO command, don't include the colon in the command. You must, however, begin the label's line with a colon. When DOS encounters the colon, it knows the line contains a label instead of a DOS command. DOS does not care if your label name appears in the upper or lower case. The following batch file successfully executes, even though the GOTO command uses a lowercase label name:

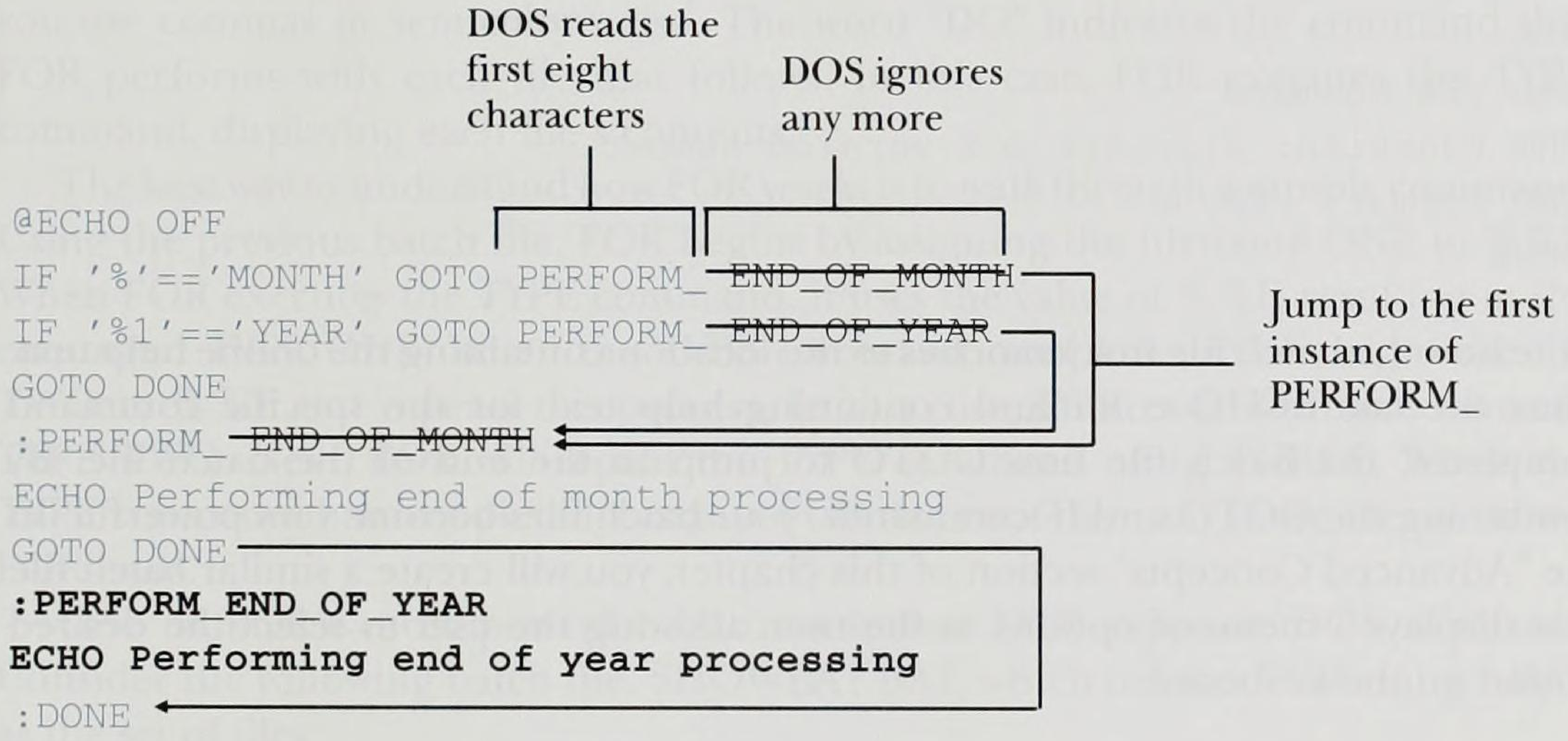
```
@ECHO OFF
VER
GOTO done
DIR
:DONE
```

If you misspell a label name and GOTO can't find a match, the batch file will end immediately, and DOS will display the "Label not found" error message. DOS will not execute any batch file commands that follow an unfound GOTO.

Although batch label names can be virtually any length, DOS only uses the first eight characters of a label name. As a result, if your batch file uses two label names whose first eight characters are the same, DOS can't tell the difference between the two labels—GOTO will branch to the closest label. For example, the following batch file uses the labels PERFORM_END_OF_MONTH and PERFORM_END_OF_YEAR. If the value of %1 is MONTH, the batch file is intended to branch to PERFORM_END_OF_MONTH. Likewise, if %1 is YEAR, the batch file should branch to PERFORM_END_OF_YEAR.

```
@ECHO OFF
IF '%1'=='MONTH' GOTO PERFORM_END_OF_MONTH
IF '%1'=='YEAR' GOTO PERFORM_END_OF_YEAR
GOTO DONE
:PERFORM_END_OF_MONTH
ECHO Performing end of month processing
GOTO DONE
:PERFORM_END_OF_YEAR
ECHO Performing end of year processing
:DONE
```

The batch file doesn't work, however. Because DOS only uses the first eight letters of a label name, it can't tell the difference between PERFORM_END_OF_MONTH and PERFORM_END_OF_YEAR. Regardless of whether %1 contains "MONTH" or "YEAR," the batch file branches to the first occurrence of "PERFORM_" with unintended results. The following illustration shows this process.



```
@ECHO OFF
IF '%'=='MONTH' GOTO PERFORM_~~END_OF_MONTH~~
IF '%1'=='YEAR' GOTO PERFORM_~~END_OF_YEAR~~
GOTO DONE
:PERFORM_~~END_OF_MONTH~~
ECHO Performing end of month processing
GOTO DONE
:PERFORM_END_OF_YEAR
ECHO Performing end of year processing
:DONE
```

DOS reads the first eight characters
DOS ignores any more
Jump to the first instance of PERFORM_

The previous batch file was the first to combine the IF and GOTO commands. When a condition is true and you want a batch file to perform several commands, you group the commands at a specific location in the batch file and then let the batch file branch to them. For example, assume you are creating a batch file named HELPDOS.BAT that contains online help for key DOS commands. To use the batch file, invoke HELPDOS.BAT with a command name, as shown in the following.

```
C:\BATCH> HELPDOS DIR <ENTER>
```

The batch file examines %1 and branches to a corresponding location within the batch file containing ECHO commands that describe the command. The following batch file implements the online help for the CLS, DIR, and VER commands.

```
@ECHO OFF
IF '%1'=='CLS' GOTO CLS
IF '%1'=='DIR' GOTO DIR
IF '%1'=='VER' GOTO VER
GOTO DONE
:CLS
ECHO CLS Command
ECHO Function: Clears the screen display.
ECHO Format: CLS
GOTO DONE
:DIR
ECHO DIR Command
ECHO Function: Displays a directory listing.
ECHO Format: DIR [file name] [switches]
GOTO DONE
:VER
ECHO VER Command
ECHO Function: Displays DOS version number.
ECHO Format: VER
:DONE
```

Note how the batch file first branches to the location containing the online help text. After the last ECHO command containing help text for the specific command completes, the batch file uses GOTO to jump to the end of the batch file. By combining the GOTO and IF commands, your batch files become very powerful. In the "Advanced Concepts" section of this chapter, you will create a similar batch file that displays a menu of options to the user, allowing the user to select the desired option on the keyboard.

Repeating a Command for a Set of Files

As you have learned, not all DOS commands support wildcards. For example, if you invoke TYPE using a wildcard, the command will fail.

```
C:\> TYPE  *.BAT <ENTER>
Invalid filename or file not found
```

By creating a batch file that uses the FOR command, however, you can provide the wildcard support you desire. FOR is an internal command that lets you repeat a specific command for a set of files. Although you can execute FOR from the DOS prompt, FOR is most commonly used in DOS batch files. In general, you execute FOR with a set of filenames and a command you want FOR to execute with each file. FOR will execute the command for the first file. When the command completes, FOR will execute the command for the second file, followed by the third, and so on for every file in the set. Consider the following batch file TYPE3.BAT, which uses FOR to display the contents of the files ONE, TWO, and THREE.

```
@ECHO OFF
FOR  %%F  IN  (ONE TWO THREE)  DO  TYPE  %%F
```

The characters %%F represent the FOR command's variable. As you just read, FOR repeats a command for a set of files. Each time FOR repeats the command, FOR assigns a new filename to its variable. FOR variable names can only be one character long. The word "IN" indicates that the set of files follows. You must place the set of files within parentheses. This example separates the files with spaces, but FOR lets you use commas or semicolons also. The word "DO" indicates the command that FOR performs with each file that follows. In this case, FOR executes the TYPE command, displaying each file's contents.

The best way to understand how FOR works is to walk through a sample command. Using the previous batch file, FOR begins by assigning the filename ONE to %%F. When FOR executes the TYPE command, it uses the value of %%F, resulting in the command TYPE ONE. When the TYPE command completes, FOR assigns the file TWO to %%F and repeats this process. In this case, FOR executes the command TYPE TWO. Next, FOR will repeat the TYPE command using THREE. When the TYPE command has completed, because FOR does not have any more files in its set, FOR ends.

One of FOR's most powerful features is that FOR supports DOS wildcards. Consider the following batch file, SHOWBAT.BAT, which invokes FOR using *.BAT as the set of files:

```
@ECHO OFF
FOR %%F  IN  (*.BAT)  DO TYPE  %%F
```

When FOR encounters a wildcard in its set of files, FOR expands the wildcard into the complete set of matching files.

Taking this concept one step further, the batch file shown in the next example, TYPEALL.BAT, uses the batch parameter %1 to define the set of files.

```
@ECHO OFF
FOR %%F  IN  (%1)  DO TYPE  %%F
```

If you invoke TYPEALL.BAT with a filename, such as AUTOEXEC.BAT, the batch file will display the file's contents. If you invoke TYPEALL.BAT using a wildcard, the batch file will display all matching files.

```
C:\BATCH> TYPEALL  *.BAT <ENTER>
```

Last, if you invoke TYPEALL.BAT without specifying a filename, FOR encounters an empty set as soon as it starts. As a result, it ends without executing the TYPE command.

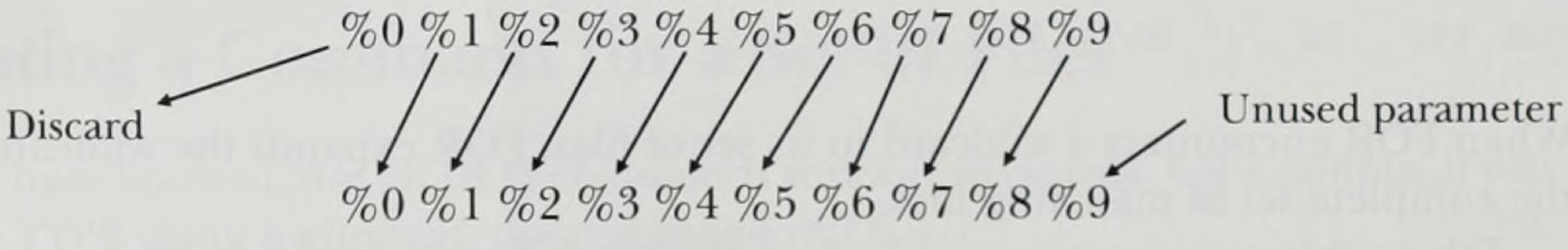
```
C:\BATCH>  TYPEALL <ENTER>
C:\BATCH>
```

You can execute the FOR command from the DOS prompt. When you execute FOR from the DOS prompt, only precede the variable name with a single percent sign, as shown here:

```
C:\> FOR  %F  IN  (*.BAT)  DO  TYPE  %F <ENTER>
```

Accessing More than Nine Batch Parameters

You learned in Chapter 16 that DOS lets you pass information to your batch files using batch parameters. DOS assigns the parameters to the symbols %1 through %9. If the user specified more than nine parameters, the batch files that were presented in Chapter 16 ignored the extras. To help your batch files use all the information the user includes in the command line, DOS provides the SHIFT command. In general, SHIFT moves each parameter one position to the left, discarding %0's value and moving the first unused parameter into %9, as shown here:



If no unused parameters are present, SHIFT assigns the null string to %9.

The following batch file, SHIFTONE.BAT, uses the SHIFT command to move the batch parameters one position to the left:

```
@ECHO OFF
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

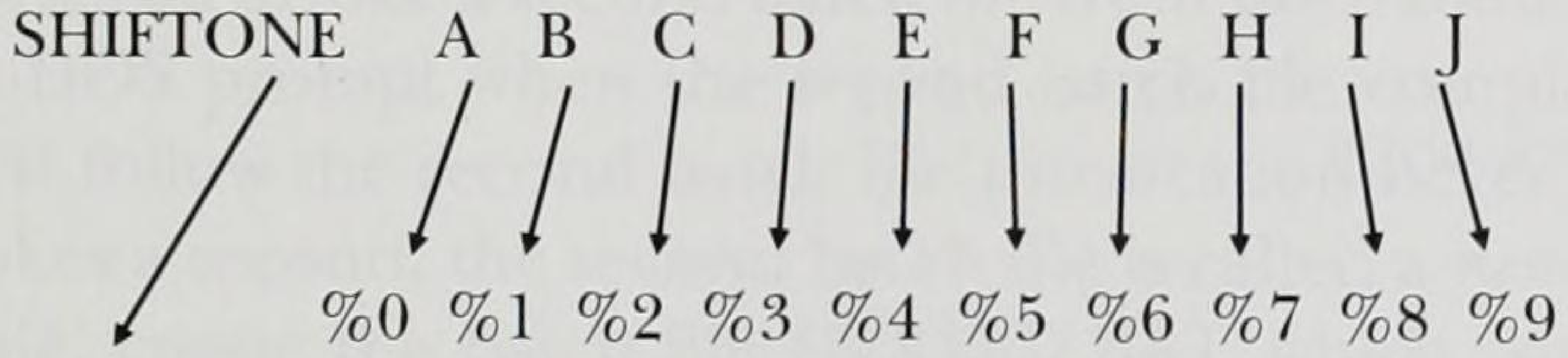
If you invoke the batch file with the parameters A, B, and C, your screen will display the following:

```
C:\BATCH> SHIFTONE A B C <ENTER>
SHIFTONE A B C
A B C
```

As you can see, each letter moved one position left with "A" now occupying the position that "SHIFTONE" held. If you invoke the batch file with the letters A through L, your screen will display the following:

```
C:\BATCH> SHIFTONE A B C D E F G H I J K L <ENTER>
SHIFTONE A B C D E F G H I
A B C D E F G H I J
```

As you can see, the batch file's name has been shifted out of %0 and the letter J has been shifted into %9, as shown here:



In a similar way, the batch file SHIFTTWO.BAT uses ECHO to display the before, middle, and after of two SHIFT commands.

```
@ECHO OFF
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

As above, if you invoke this batch file with the letters A through L, your screen will display the following:

```
C:\BATCH> SHIFTTWO A B C D E F G H I J K L <ENTER>
SHIFTTWO A B C D E F G H I
A B C D E F G H I J
B C D E F G H I J K
```

For SHIFT to really be useful, your batch files need a way to use each parameter individually. The following batch file, SHIFTALL.BAT, combines the SHIFT and GOTO commands. The batch file examines the value of %1. If %1 contains the null string, the batch file ends. Otherwise, it echoes %1's value and performs a SHIFT. Using GOTO, the batch file loops back to the start and repeats this process. If you invoke the batch file with the letters A through Z, the batch file will display each letter on its own line.

```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
ECHO %1
SHIFT
GOTO LOOP
:DONE
```

A more useful version of this batch file might use the TYPE command to display the contents of multiple filenames, like this:

```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
TYPE %1
SHIFT
GOTO LOOP
:DONE
```

In this case, you can invoke the new TYPEALL.BAT file with several filenames, as shown here:

```
C:\BATCH> TYPEALL \CONFIG.SYS \AUTOEXEC.BAT SHIFTONE.BAT <ENTER>
```

Taking this batch file one step further, you can include the FOR command, discussed in the previous section, to expand any wildcard characters the command line may contain, such as the following:

```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
FOR %%F IN (%1) DO TYPE %%F
SHIFT
GOTO LOOP
:DONE
```

The batch file now supports multiple filenames and DOS wildcards.

```
C:\BATCH> TYPEALL \CONFIG.SYS  *.BAT SHIFTONE.BAT <ENTER>
```

Invoking One Batch File from Within Another

DOS batch files contain the names of one or more commands. When you invoke the batch file from the DOS prompt, DOS executes all the commands the batch file contains. All the commands in the batch file presented so far have been DOS commands. A batch file, can, however, execute another batch file as one of its commands. However, as you will see, executing one batch file from within another isn't as easy as simply including the second batch file's name. Depending on your version of DOS, to invoke a second batch file, your batch file must use the CALL command or COMMAND /C.

By default, when DOS ends a batch file, control returns to the DOS prompt. Unfortunately, if you invoke a second batch file from the middle of a batch file, DOS redisplays the DOS prompt when the second batch file completes. As a result the commands that follow the second batch file's invocation never execute. When one batch file invokes a second, the second batch file is called a *nested batch file.*

For example, create the batch file NESTED.BAT, which contains the following ECHO commands:

```
ECHO  In the batch file NESTED.BAT
ECHO  All done here
```

Next, create the batch file NOEND.BAT, which contains the following:

```
@ECHO OFF
ECHO About to call the nested batch file
NESTED
ECHO Back from the nested batch file
ECHO Done!
```

As you can see, the third line of the batch file executes NESTED.BAT. When you invoke NOEND.BAT, your screen displays the following:

```
C:\BATCH> NOEND <ENTER>
About to call the nested batch file
In the batch file NESTED.BAT
All done here
C:\BATCH>
```

You can see that the NOEND.BAT batch file invokes NESTED.BAT. However, when NESTED.BAT ends, DOS redisplays its prompt. As a result, the last two lines in NOEND.BAT never execute.

To get a nested batch file to return control to the batch file that called it, you must invoke the nested batch file using CALL or COMMAND /C. In DOS 3.3 or later, CALL is an internal command that lets one batch file invoke (or call) another. Using DOS 3.3 or later, change the batch file NOEND.BAT to the following:

```
@ECHO OFF
ECHO About to call the nested batch file
CALL NESTED
ECHO Back from the nested batch file
ECHO Done!
```

If you are using a DOS version earlier than 3.3, change the NOEND.BAT file to use COMMAND /C, as shown here:

```
@ECHO OFF
ECHO About to call the nested batch file
COMMAND /C NESTED
ECHO Back from the nested batch file
ECHO Done!
```

In either case, once you change NOEND.BAT, DOS will execute the nested batch file and then return control to the original batch file, letting the batch file complete. Your screen will then display the following:

```
C:\BATCH> NOEND <ENTER>
About to call the nested batch file
In the batch file NESTED.BAT
All done here
Back from the nested batch file
Done!
```

The only time you don't need to use CALL or COMMAND /C to invoke a nested batch file is when you invoke the nested batch file as the last command. The nested batch file HELPDOS is the last command called in the following batch file:

```
@ECHO OFF
PATH C:\DOS
PROMPT $P$G
HELPDOS
```

You don't care that DOS won't return control in this case, because no commands follow the nested batch file. As a general rule, however, use COMMAND/C or CALL to invoke nested batch files.

SUMMARY

GOTO

The GOTO command directs DOS to jump from one location in your batch file directly to another. The format of the GOTO command is

```
GOTO LABELNAME
```

A *label* is a unique name that appears on its own line in your batch file. The first character of the line must be a colon. When DOS encounters the colon, DOS knows the line contains a label and does not try to execute the line as a command. If your batch file does not have a label whose name matches the label specified in the GOTO command (case doesn't matter), DOS will end the batch file immediately and display an error message. Although your label names can essentially contain any number of characters, DOS only reads the first eight. If two labels in your batch file have the same first eight characters, your batch file will not behave as intended.

FOR

The FOR command lets you repeat a specific command for a set of files. FOR's format in batch files is as follows:

```
FOR %%F IN (FILES) DO DOS_COMMAND
```

The FOR command assigns a file from the set of files to its variable %%F and performs the command specified. When the command completes, FOR assigns the next file in the set to the variable. This process repeats until FOR

SUMMARY

uses every file in the set. If the set contains a DOS wildcard, FOR expands the wildcard into the matching filenames. DOS lets you invoke FOR from the DOS prompt as well as within batch files. If you invoke FOR from the DOS prompt, only precede FOR's variable name with a single percent sign.

SHIFT

The SHIFT command provides your batch file with access to more than the nine batch parameters DOS assigns to %1 through %9. When your batch file executes SHIFT, DOS moves each batch parameter one position to the left, discarding the value of %0 and shifting into %9 the first unused parameter. If no unused parameters remain, DOS assigns %9 the null string. The format of the SHIFT command is simply SHIFT.

Nested Batch Files

A nested batch file is a batch file invoked by another batch file. When a batch file invokes a second batch file, you need to use the CALL command or COMMAND /C. If you don't, DOS will return control to the DOS prompt when the nested batch file completes and the remaining commands in the original batch file never execute. CALL requires DOS 3.3 or later. The format of the CALL command is as follows:

```
CALL NESTED OPTIONAL_PARAMETERS
```

Users of DOS versions earlier than 3.3 must use the COMMAND /C command, whose format is as follows:

```
COMMAND /C NESTED  OPTIONAL_PARAMETERS
```

Hands On

If your batch file misspells a label and GOTO can't find a match, DOS immediately ends the batch file execution and displays the "Bad label" error message. Verify this by creating the batch file BADLABEL.BAT, as shown here:

```
@ECHO OFF
GOTO DONE
DATE
TIME
```

Execute BADLABEL from the DOS prompt:

```
C:\BATCH> BADLABEL <ENTER>
```

Since there is no line labeled ":DONE", the batch file ends immediately after it encounters GOTO, never executing DATE or TIME.

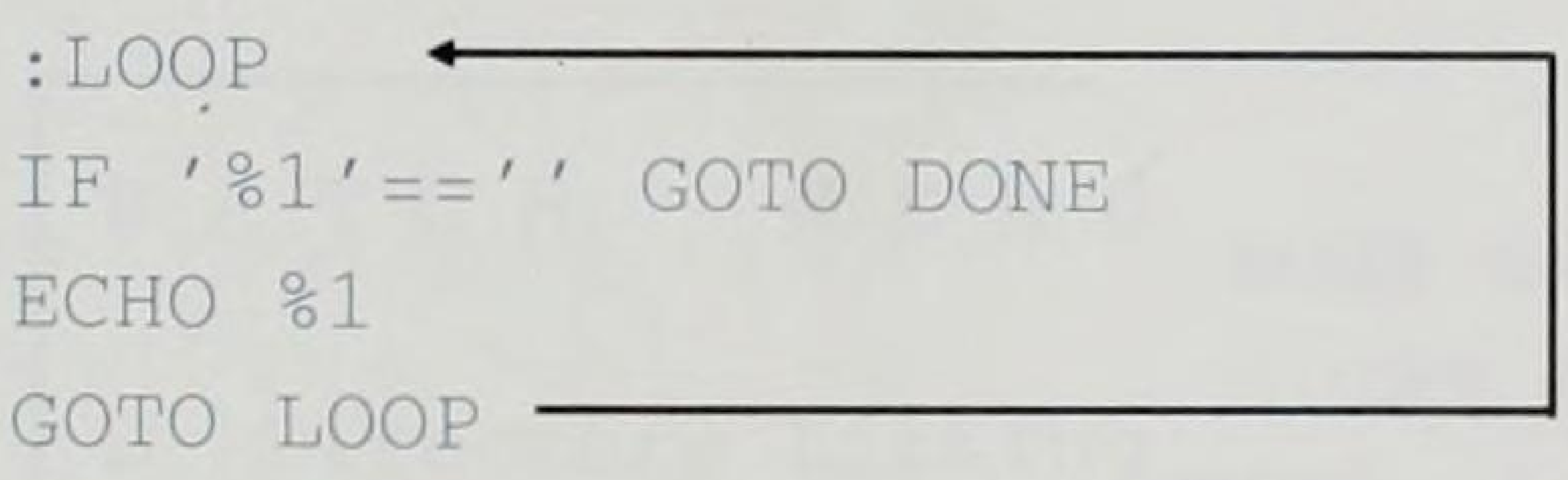
When you use GOTO to create a loop, you need to make sure you give the batch file a way to end. The following batch file, INFINITE.BAT, uses GOTO, SHIFT, and ECHO to display each of the batch file's command-line parameters. Unfortunately, the batch file has the SHIFT and GOTO commands out of order.

```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
ECHO %1
GOTO LOOP
SHIFT
:DONE
```

Invoke INFINITE.BAT with the parameters A, B, and C:

```
C:\BATCH> INFINITE  A  B  C <ENTER>
```

Rather than displaying the parameters on individual lines, the batch file displays "A," jumps up to LOOP, and displays "A" again. Because GOTO is before SHIFT, the parameters never get shifted; %1 is never assigned the null string; and the file, therefore, never goes to DONE. The batch file enters a loop that never ends, an *infinite loop.*



```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
ECHO %1
GOTO LOOP
```

In DOS 5 or later, the DIR command, using the /B switch, will display each filename on its own line.

```
C:\BATCH> DIR  /B <ENTER>
```

If you are using an earlier DOS version than 5, you can create the batch file DIR1LINE.BAT, shown here, to achieve the same effect:

```
@ECHO OFF
FOR %%F IN (%1) DO ECHO %%F
```

Using DIR1LINE.BAT, you can display a directory for a single file or use a wildcard combination:

```
C:\BATCH> DIR1LINE  *.BAT <ENTER>
```

In this case, FOR expands *.BAT into the set of matching files and then uses ECHO to display each filename on its own line.

Review

1. Why do batch label name lines begin with a colon (:)?
2. Use the GOTO command to create the batch file MULTIDEL.BAT, shown here, that lets you delete one or more files:

```
C:\BATCH>  MULTIDEL  ONE  TWO  THREE <ENTER>
```

3. When must you use the CALL command or COMMAND /C? Why?
4. The IF command does not consider the uppercase letters the same as lowercase. The following batch file, COLORS.BAT, compares %1 to three colors RED, BLUE, and GREEN. If the user types the color in lowercase letters, the batch file doesn't match a color.

```
@ECHO OFF
IF '%1'=='BLUE' ECHO The color is BLUE
IF '%1'=='RED' ECHO The color is RED
IF '%1'=='GREEN' ECHO The color is GREEN
```

Use the FOR command to have the batch file support the entries RED, Red, red, BLUE, Blue, blue, GREEN, Green, and green.

Answers

1. When DOS encounters a colon as the first character in a batch file line, DOS knows the line contains a label and not a command. As a result, DOS makes a note of the label's position within the batch file, and DOS does not try to execute the line as a command.

2. One way to solve this is

```
@ECHO OFF
:LOOP
IF '%1'=='' GOTO DONE
FOR %%F IN (%1) DO DEL %%F
SHIFT
GOTO LOOP
:DONE
```

3. You must use either the CALL command or COMMAND /C to invoke another batch file from within the middle of a batch file. By default, when DOS finishes a batch file, DOS returns control to the DOS prompt. Unfortunately, if your batch file invokes another from within the middle of a list of commands and DOS returns control to the DOS prompt when the nested batch file completes, the commands that follow the batch file invocation will never execute. If you are using DOS 3.3 or later, use the CALL command; otherwise, use COMMAND /C.

4. The new batch file would be

```
@ECHO OFF
FOR %%F IN (BLUE Blue blue) DO IF '%1'=='%%F' ECHO BLUE
FOR %%F IN (GREEN Green green) DO IF '%1'=='%%F' ECHO GREEN
FOR %%F IN (RED Red red) DO IF '%1'=='%%F' ECHO RED
```

Advanced Concepts

One of the most common uses of GOTO is to create a menu-driven batch file. For example, the batch file MENU.BAT displays the following menu and waits for the user to press a key.

```
B A T C H   M E N U

D  Directory Listing
T  Set or Display TIME
V  Display DOS Version Number
Q  Quit to DOS
```

The batch file uses the command SCANCODE.COM which you will create in Chapter 29 using DEBUG. SCANCODE returns an exit status value that corresponds to a key pressed.

```
@ECHO OFF
:MAIN_LOOP
CLS
ECHO B A T C H    M E N U
ECHO.
ECHO D  Directory Listing
ECHO T  Set or Display TIME
ECHO V  Display DOS Version Number
ECHO Q  Quit to DOS
:SCAN_LOOP
SCANCODE
IF ERRORLEVEL 32 IF NOT ERRORLEVEL 33 GOTO DIR
IF ERRORLEVEL 47 IF NOT ERRORLEVEL 48 GOTO TIME
IF ERRORLEVEL 20 IF NOT ERRORLEVEL 21 GOTO VER
IF ERRORLEVEL 16 IF NOT ERRORLEVEL 17 GOTO QUIT
GOTO SCAN_LOOP
:DIR
DIR
PAUSE
GOTO MAIN_LOOP
:VER
VER
PAUSE
GOTO MAIN_LOOP
:TIME
TIME
PAUSE
GOTO MAIN_LOOP
:QUIT
```

The batch file displays the menu and uses SCANCODE to wait for the user to press a key. If the key corresponds to one of the menu options, the batch file executes the corresponding commands and then uses GOTO to loop back to the menu display repeating this process. If the user presses an invalid key, the batch file uses GOTO to loop back to SCANCODE to get a valid key.

If you are using DOS 6, the following batch file displays the same menu, using the CHOICE command to get the user's response:

```
@ECHO OFF
:MAIN_LOOP
CLS
```

```
ECHO B A T C H    M E N U
ECHO.
ECHO D  Directory Listing
ECHO T  Set or Display TIME
ECHO V  Display DOS Version Number
ECHO Q  Quit to DOS
CHOICE /C:DTVQ Enter Choice:
IF NOT ERRORLEVEL 1 GOTO QUIT
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO DIR
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO TIME
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO VER
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO QUIT
:DIR
DIR
PAUSE
GOTO MAIN_LOOP
:VER
VER
PAUSE
GOTO MAIN_LOOP
:TIME
TIME
PAUSE
GOTO MAIN_LOOP
:QUIT
```

Saving and Restoring the Current Directory

As they increase in complexity, your batch files may change the current drive or directory or even create files during their processing. As a rule, your batch files should clean up after themselves, deleting their temporary files and restoring the original drive and directory.

If you are using DOS 3.3 or later, the following batch file SAVEDIR.BAT creates an environment entry named ORIG_DIR that contains the original directory. To use this batch file, you must create the file ORIG_DIR.DAT that contains the SET command, shown here:

```
C:\BATCH> COPY CON ORIG_DIR.DAT <ENTER>
SET ORIG_DIR=^Z <ENTER>
        1 File(s) copied
```

You must place the end-of-file character (CTRL-Z) immediately next to the equal sign.

You learned in Chapter 10 that if you invoke CHDIR without a directory name, CHDIR displays the current directory. Using the DOS output redirection operator

(>>), discussed in Chapter 20, you can append CHDIR's output to the SET command, creating a batch file that saves the current directory, as shown here:

```
@ECHO OFF
COPY \BATCH\ORIG_DIR.DAT  SAVEIT.BAT
CHDIR >> SAVEIT.BAT
CALL SAVEIT
DEL SAVEIT.BAT
```

The batch file copies your template file containing the partial SET command to a batch file named SAVEIT.BAT. Next, the batch file appends to SAVEIT.BAT's SET command, the name of the current directory, as redirected by CHDIR. Assuming your current directory was DOS on drive C, SAVEIT.BAT's SET command will now be SET ORIG_DIR=C :\DOS. Next, the batch file uses CALL to invoke SAVEIT.BAT, which in turn uses SET to define the ORIG_DIR environment entry. Because the batch file SAVEIT.BAT has performed its processing, the batch file deletes it, cleaning up after itself.

The following batch file, LISTTREE.BAT, invokes SAVEDIR.BAT using the CALL command. Next, the batch file selects the root directory and invokes the TREE command. When TREE completes, the batch file uses the named parameter, ORIG_DIR, from above, to restore the original directory.

```
@ECHO OFF
CALL SAVEDIR
CHDIR \
TREE
CHDIR %ORIG_DIR%
SET ORIG_DIR=
```

As you create your batch files, use techniques such as SAVEDIR to reduce the side effects of your more complex batch files.

Key Terms

Branch To change the order in which a program or batch file is executed. In DOS batch files, branching (jumping) is accomplished through use of the GOTO command and a label.

Label A name in a batch file that GOTO can use as a marker to jump to when called. The first character on the line where the label appears must be a colon. This tells DOS that what is on the line is a label, not a command.

Nested batch file A second batch file invoked by a batch file. To invoke nested batch files, you need to use the CALL command or COMMAND /C.

Chapter 18

Automatic System Startup Commands with AUTOEXEC.BAT

You have seen that DOS batch files save you time and keystrokes while eliminating your need to memorize long or difficult commands. Using batch files, you can combine several related commands into one. As you work with DOS, you may find that you always perform several important commands each time your system starts, for example, defining the command path and system prompt or invoking the FASTOPEN command to improve your file access time. In this case, by placing the PATH, PROMPT, and FASTOPEN commands into a batch file, you can quickly perform all three steps with a single command.

You still have to call your setup batch file each time your system starts and it would be convenient if you could tell DOS to do it for you. As it turns out, you can. Each time DOS starts, it searches your startup disk's root directory for a special batch file named AUTOEXEC.BAT. If this batch file exists, DOS executes all the commands in it. In other words, DOS automatically executes your system startup commands.

If AUTOEXEC.BAT does not exist in the root directory, DOS executes the DATE and TIME commands. DOS uses the batch file name AUTOEXEC.BAT because it automatically executes the commands in the batch file. In this chapter you will examine several commands that users typically place in their AUTOEXEC.BAT file.

Common AUTOEXEC.BAT Commands

DOS provides the AUTOEXEC.BAT batch file to help you execute the commands you need to run each time your system starts. Table 18-1 lists several DOS commands users commonly place in their AUTOEXEC.BAT file.

PATH Defines Your Command Path

You learned in Chapter 11 that the PATH command lets you define the list of directories DOS searches for your external commands and batch files whenever it fails to locate the command in the current directory. At the very minimum, your command path should include the DOS directory, which contains your external DOS commands.

```
PATH C:\DOS
```

Depending on the applications you use on a regular basis, you may include additional directories, such as WINDOWS, 123, and WP51 (for WordPerfect), as shown here:

```
PATH C:\DOS;C:\WINDOWS;C:\123;C:\WP51
```

Command	Function
PATH	Defines the DOS command path
PROMPT	Defines the DOS prompt
FASTOPEN	Improves file access performance (DOS 3.3 or later)
PRINT	Installs the PRINT command's memory-resident software
SET TEMP	Defines the disk and directory in which DOS creates its temporary files (DOS 5 or later)
SET DIRCMD	Defines the default command-line switches for the DIR command (DOS 5 or later)
SHARE	Provides additional support for large disks in DOS version 4, and file locking in network environments
DOSSHELL	Invokes the DOS shell user interface (DOS 4 or later)
WIN	Starts the Windows graphical user interface
VSAFE	Starts the DOS 6 virus protection software
SMARTDRV	Loads disk caching software (DOS 6)

Table 18-1. *Common AUTOEXEC.BAT File Commands*

By placing the PATH command in your AUTOEXEC.BAT file, DOS automatically sets up your command path each time your system starts. Only place directories that contain your commonly used commands in the command path. If you include many directories, DOS will waste considerable time searching directories that are unlikely to contain your current command.

PROMPT Defines Your System Prompt

In Chapter 10 you used the following PROMPT command to display the current drive and directory within your system's prompt:

```
C> PROMPT  $P$G <ENTER>
C:\>
```

Although you may choose to further customize your prompt, I strongly recommend that your prompt include the $P and $G entries to display the current drive and directory information. By placing the above PROMPT command in AUTOEXEC.BAT, you can always easily determine the current directory because it will be displayed to you with each DOS prompt.

FASTOPEN Decreases File Access Time

If you are using DOS version 3.3 or later, the FASTOPEN command, discussed in Chapter 11, can improve your performance by reducing the amount of time your computer spends searching your disk for commonly used files.

By default, each time you run a program or open a file, whether the file is a letter, report, or work sheet, DOS must search your disk to locate the file. Although your hard disk is fast, it is still mechanical, and thus much slower than your computer's electronic memory. FASTOPEN directs DOS to store, in your computer's fast electronic memory, the name and disk location of files as you access them. Each time you access a file, DOS searches FASTOPEN's list of recently used files to see if the file is among them. If the file is in the list, DOS immediately knows the file's disk location and does not have to send the disk drive head out searching the disk for it. If the file is not present in the list, DOS must perform the slower, normal disk search.

The FASTOPEN command will not improve the access time for every file you access, only for the files you access more than once. The following FASTOPEN command directs DOS to track the names and the disk locations of up to 50 recently used files:

```
FASTOPEN  C:=50
```

You may specify FASTOPEN for more than one disk, as in the next example.

```
FASTOPEN  C:=50  D:=50
```

If you are using DOS 3.3 or later, you should include FASTOPEN in your AUTOEXEC.BAT file.

Installing a High-Performance Print Queue

Chapter 14 discusses the PRINT command in detail and, in the "Advanced Concepts" section, presents several command-line switches that you can include the first time you invoke PRINT, which improve its performance. If you use the PRINT command on a regular basis, place the following command in your AUTOEXEC.BAT file:

```
PRINT  /B:4096  /D:PRN  /M:32  /U:8
```

The /B entry increases the print buffer from 512 bytes to 4096 bytes, meaning that DOS can give your printer bigger amounts of data at a time and get on with its other business. The /D entry specifies which port to use. If your printer attaches to a serial port, replace PRN with the serial port name, such as COM1. The /M entry increases the amount of time PRINT can control the CPU, from 2 ticks (1/10 second) to 32 ticks (1.4 seconds). The /U switch increases how long PRINT should wait (for an older, slower printer), from the default of 2 to 8 ticks. If you don't need some of these switches, you won't need to include them—if you don't use PRINT to print your files, you don't need to include this command in your AUTOEXEC.BAT file at all. Because PRINT only lets you specify these switches the first time you invoke it, placing the PRINT command in AUTOEXEC.BAT ensures that the correct switches are in use each time your system starts.

Informing DOS Where to Create Its Temporary Files

In Chapter 21 you will examine the DOS *pipe operator,* which lets one command's output become another command's input. For example, the following command displays a sorted directory listing by *piping* the output of the DIR command into the SORT command.

```
C:\> DIR  |  SORT <ENTER>
```

To implement the pipe, DOS needs to create temporary files. By default, DOS creates these temporary files in your disk's root directory. Unfortunately, if your disk does not have enough disk space for the temporary files, the pipe operation will fail.

In DOS 5 or later, you can tell DOS where to create these temporary files with the DOS environment entry, TEMP. If you create an electronic-memory RAM drive (Chapter 24 discusses RAM drives in detail) with the RAMDRIVE.SYS device driver and then assign TEMP to point to the RAM drive, DOS can create the temporary files faster, which further improves your system performance. The following SET command assumes you have a fast RAM drive named E:

```
SET TEMP=E:
```

The TEMP environment entry is only used by DOS 5 or later. If you are using a different DOS version, you do not need to include such a SET command in your AUTOEXEC.BAT unless you know it is required by your applications software.

Specifying the DOS DIR Command Attributes

Chapter 4 examined the DIR command. At that time, you learned DIR displays a file's base name, extension, size, and date and time stamps. By default, DIR does not display the files in sorted order. If you are using DOS 5 or later, however, DIR lets you specify a desired sort order.

The "Advanced Concepts" section of Chapter 4 describes the /O, /A, /S,/L, and B/ switches that you can use to customize your DOS 5 or later directory listing. As you experiment with these switches, you will eventually find a combination that you like best. Rather than forcing you to remember and type these switches each time you invoke DIR, DOS lets you define the DIRCMD environment entry. Each time you invoke DIR to display a directory listing under DOS 5 or later, DIR searches the environment for an entry in the form DIRCMD=*SWITCHES*. If DIR finds the entry, DIR uses the switches specified. Placed in your AUTOEXEC.BAT, the DIRCMD entry lets you define the default format of your directory listing.

The following SET command creates a default format in which DIR is sorted by name and displayed in the lower case:

```
SET  DIRCMD=/O:N/L
```

Like TEMP, DIRCMD is only available in DOS 5 or later.

Invoking the DOS Shell

DOS, in versions 4 and later, provides a user-friendly command shell that lets you work in a menu-driven environment. Many users who prefer to work from the DOS shell, as opposed to the command line, include the DOSSHELL command as the last command in their AUTOEXEC.BAT. In DOS 4, DOSSHELL is a batch file and not an actual DOS command, so you should make DOSSHELL the last command in your AUTOEXEC.BAT file. As you learned in Chapter 17, when you invoke one batch file from within another without using CALL or COMMAND /C, the first batch file never regains control. If DOSSHELL is the last command in the batch file, you don't have to worry about AUTOEXEC.BAT regaining control; all of its work is already done.

SHARE

In DOS 4, if you are using a hard disk partition larger than 32 megabytes, you need to install the SHARE command. In DOS 4, SHARE ensures that older programs using file control blocks (FCBs) don't destroy information on your disk. File control blocks are not compatible with the larger partition size. Therefore, many DOS 4 users place a SHARE command in AUTOEXEC.BAT to ensure the larger disk is supported when your system starts. However, for better memory utilization, you should use the CONFIG.SYS INSTALL= entry discussed in Chapter 24.

Starting Microsoft Windows

If you have installed Microsoft Windows on your system, you can direct DOS to automatically invoke Windows each time your system starts by including the WIN command in your AUTOEXEC.BAT file. By invoking Windows in this way, you never have to worry about issuing commands from the DOS prompt.

Installing VIRUS Protection

If you are using DOS 6, the VSAFE command lets you check your files for viruses. If you frequently download files from computer bulletin boards or exchange disks with other users, you might want to consider invoking the VSAFE command from within your AUTOEXEC.BAT file. In that way, VSAFE can scan your disk for viruses each time your system starts.

SMARTDRV Disk Caching

Chapter 31 discusses SMARTDRV's disk-caching software and how it improves your system performance. Prior to DOS 6, users installed the disk caching software using the SMARTDRV.SYS device driver. Beginning with DOS 6 (and Windows 3.1), SMARTDRV became a command (SMARTDRV.EXE). If you are using DOS 6, or if you have a version of SMARTDRV.EXE that was provided with Windows, you should invoke SMARTDRV from within your AUTOEXEC.BAT file.

A Sample AUTOEXEC.BAT

Depending on your own requirements, the actual commands that appear in your AUTOEXEC.BAT file may differ from those shown here. The purpose of this file is to give you a sample format for creating your own AUTOEXEC.BAT file.

```
@ECHO  OFF
PATH  C:\DOS
PROMPT  $P$G
FASTOPEN  C:=50
PRINT  /B:4096  /D:PRN  /M:32  /U:8
DOSSHELL
```

AUTOEXEC.BAT and CONFIG.SYS

DOS only uses the AUTOEXEC.BAT batch file once—when your system starts up. For DOS to locate AUTOEXEC.BAT, it must reside in your startup disk's root directory. In addition to AUTOEXEC.BAT, DOS uses a second user-defined file named CONFIG.SYS, which must also reside in the root. Chapter 24 discusses the CONFIG.SYS file in detail. It is important that you understand the difference between these two files.

AUTOEXEC.BAT contains one or more *DOS commands* that DOS executes when your system starts. The CONFIG.SYS file contains single-line *system configuration entries* that tell DOS such information as the number of files it may need to open at one time. CONFIG.SYS entries are not DOS commands. The following TYPE command illustrates possible CONFIG.SYS entries:

```
C:\> TYPE  \CONFIG.SYS <ENTER>
FILES=20
BUFFERS=25
DEVICE=C:\DOS\RAMDRIVE.SYS
```

These entries control how DOS configures itself in system memory, hence the name CONFIG.SYS. Use DIR to examine the files in your root directory. Your root directory should have an AUTOEXEC.BAT file and a CONFIG.SYS file. Use TYPE to display the contents of each.

AUTOEXEC.BAT will contain recognizable DOS commands. CONFIG.SYS, on the other hand, will contain single-line entries DOS uses during the startup process.

NOTE

Change AUTOEXEC.BAT with Caution

AUTOEXEC.BAT is a very powerful tool. It runs powerful commands automatically each time your system starts. You need to be very careful when you make changes to AUTOEXEC.BAT. If you make an error, you could possibly prevent DOS from being able to boot your system. As a rule, always make a copy of your AUTOEXEC.BAT file before you make any changes to it. The following command, for example, copies AUTOEXEC.BAT to a file named AUTOEXEC.SAV.

```
C:\> COPY  \AUTOEXEC.BAT  \AUTOEXEC.SAV <ENTER>
```

Should you make an errant entry or deletion to or from AUTOEXEC.BAT, you can reboot your system using a bootable DOS floppy disk and restore the original AUTOEXEC.BAT file from the saved one.

Keep in mind that the only time DOS uses AUTOEXEC.BAT is during system startup. If you make changes to AUTOEXEC.BAT, you will need to reboot to use the new AUTOEXEC.BAT.

SUMMARY

AUTOEXEC.BAT

Each time DOS starts, it searches your startup disk's root directory for a special batch file named AUTOEXEC.BAT. If it exists, DOS executes each of the commands that the batch file contains. AUTOEXEC.BAT lets you specify commands you want DOS to automatically execute at system startup. If the file does not exist, DOS executes the familiar DATE and TIME commands.

Hands On

Using the TYPE command, display the current contents of your AUTOEXEC.BAT file.

```
C:\> TYPE  \AUTOEXEC.BAT <ENTER>
```

So that you have a backup copy of the file's current contents, copy AUTOEXEC.BAT to AUTOEXEC.SAV, as shown here:

```
C:\> COPY  \AUTOEXEC.BAT  \*.SAV <ENTER>
```

Next, using your word processor or the DOS text editor, you can edit your AUTOEXEC.BAT file. If you use a word processor, as discussed in Chapter 15, you must edit batch files using ASCII or unformatted mode. Do not remove any commands from AUTOEXEC.BAT that you do not understand. If you would like, add commands to set your system prompt and define a command path. If you are using DOS 3.3 or later, include a FASTOPEN command to improve your system performance. If you are using DOS 5 or later, consider adding SET commands for TEMP and DIRCMD. For more information on these two entries, see Chapter 19.

If you make any changes to AUTOEXEC.BAT, reboot your system to put the changes into effect.

Review

1. How does AUTOEXEC.BAT differ from other batch files?
2. How does AUTOEXEC.BAT differ from CONFIG.SYS?
3. How can you distinguish between an AUTOEXEC.BAT entry and a CONFIG.SYS entry?
4. What does DOS do if the AUTOEXEC.BAT file does not exist?
5. How does DOS know you have made changes to AUTOEXEC.BAT so DOS can put the changes into effect?

Answers

1. Like all batch files, AUTOEXEC.BAT contains a list of DOS commands. AUTOEXEC.BAT is special because each time your system starts, DOS automatically searches your root directory for AUTOEXEC.BAT. If DOS locates the file, DOS executes all of the commands AUTOEXEC.BAT contains. If AUTOEXEC.BAT does not exist, DOS executes the familiar DATE and TIME commands.
2. DOS uses the files AUTOEXEC.BAT and CONFIG.SYS during system startup. However, DOS uses the files very differently. CONFIG.SYS is a user-defined file that contains single-line entries that let you customize DOS. Common uses of CONFIG.SYS are specifying the number of files DOS can open at one time, installing disk input and output buffers in memory to improve your system performance, and installing device-driver software for RAM drives or unique hardware devices. AUTOEXEC.BAT, on the other hand, contains a list of commands you want DOS to execute each time your system starts. Common

uses of AUTOEXEC.BAT include defining the command path, setting the system prompt, and invoking the DOSSHELL user interface.

3. AUTOEXEC.BAT entries are DOS commands. CONFIG.SYS entries are system configuration values.
4. If the file AUTOEXEC.BAT does not reside in the startup disk's root directory, DOS executes the DATE and TIME commands.
5. DOS does not know. The only time DOS uses AUTOEXEC.BAT is during system startup. If you make changes to AUTOEXEC.BAT, you will probably want to restart your computer. In some cases, you can simply invoke the new AUTOEXEC.BAT commands individually from the DOS prompt to achieve the same results.

Advanced Concepts

Many users understand that DOS executes the commands in the CONFIG.SYS and the AUTOEXEC.BAT files during system startup, but they are still not quite sure how DOS uses them and when. To better understand where AUTOEXEC.BAT and CONFIG.SYS fall into the DOS system startup process, the "Advanced Concepts" section of Chapter 2 discusses the DOS system startup process in detail. You might wish to review that section now.

Shielding AUTOEXEC.BAT from Software Installations

If you have worked with DOS for some time, you have probably experienced frustration when an application program's installation procedures overwrite your AUTOEXEC.BAT file. To prevent such a loss of your AUTOEXEC.BAT, consider creating a second batch file, perhaps called CUSTOM.BAT, that contains the commands you would typically place in your AUTOEXEC.BAT file. Next, within AUTOEXEC.BAT, only include the following command, which invokes your startup commands as desired.

```
CALL CUSTOM
```

Should a program's installation overwrite your AUTOEXEC.BAT file, you only lose this single command. If you are not using DOS version 3.3 or later, change the previous CALL command to COMMAND /C, as discussed in Chapter 17. After the installation is complete, check your AUTOEXEC.BAT file to ensure that it is consistent with CUSTOM.BAT; in other words, check that CUSTOM.BAT does not overwrite a needed command-path entry.

Disabling AUTOEXEC.BAT

If DOS does not find AUTOEXEC.BAT, it executes the familiar DATE and TIME commands. If, for some reason, you don't want DOS to do either, you can disable the execution of AUTOEXEC.BAT using the CONFIG.SYS SHELL= entry discussed in Chapter 24. The SHELL= entry tells DOS the location of the command processor COMMAND.COM. Many users use SHELL= to direct DOS to use the copy of COMMAND.COM that resides in the DOS directory; they can then remove the COMMAND.COM file from the root. Likewise, the SHELL= entry lets you increase your environment size. The following CONFIG.SYS entry directs DOS to use the COMMAND.COM file that resides in the DOS directory and to create an environment of 1024 bytes.

```
SHELL=C:\DOS\COMMAND.COM  /E:1024  /P
```

If you omit the /P switch, DOS will not execute AUTOEXEC.BAT. Likewise, DOS will not execute DATE and TIME. DOS will simply display its system prompt.

Be careful if you do not include the /P switch. If you later execute the EXIT command, discussed in Chapter 41, DOS will generate a "Bad or Missing Command Interpreter" error message, and your system will hang up. The only way to recover at that point is to reboot your computer.

Finally, if you are using DOS 6, you can prevent DOS from executing your AUTOEXEC.BAT file as well as your CONFIG.SYS entries by pressing the F5 or F8 function keys when your system starts, as discussed in Chapter 24.

Key Terms

AUTOEXEC.BAT A unique user-defined batch file whose commands DOS automatically executes each time your system starts.

CONFIG.SYS A unique user-defined file containing single-line entries (not commands) that direct DOS to configure itself in memory using the information you specify.

Chapter 19

The DOS Environment

Each time DOS starts, it reserves a section of your computer's memory it calls the *environment* to store information accessible to DOS commands and your applications. Typically, the information you store is specific to your own personal working environment, for example, the current command path and your system prompt.

In this chapter you will learn how to display the environment's current entries and how to add and remove entries. If you are using DOS 5 or later, you will also learn how to use the environment to enhance the DIR command, as well as how to specify where DOS creates its temporary files.

The SET Command

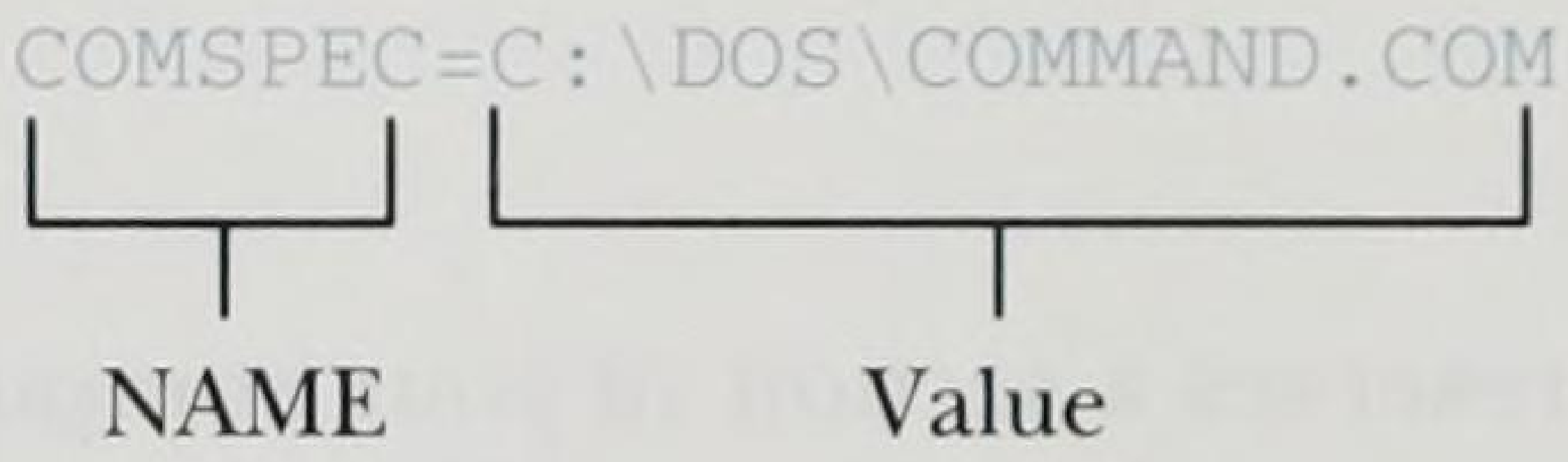
The SET command lets you display, add, and remove environment entries. From the DOS prompt, issue the following SET command to display the environment's current contents:

```
C:\> SET <ENTER>
```

Depending on your system configuration, the contents of your environment may differ from those of other users. At a minimum, your environment will probably contain a COMSPEC, PATH, and PROMPT entry:

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
```

DOS environment entries have the form *NAME=Value*. In this case, COMSPEC has the value C:\ DOS \COMMAND.COM, as shown here:



Adding an Environment Entry

In addition to letting you display the environment entries, the SET command lets you add environment entries. For example, the following creates an environment entry named BOOK and assigns it the value "DOS The Complete Reference":

```
C:\> SET  BOOK=DOS The Complete Reference <ENTER>
```

Using the SET command, you can display the environment entry's current contents, ensuring that SET added the BOOK entry.

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
BOOK=DOS The Complete Reference
```

SET always converts the entry name to uppercase letters, but leaves the entry's value in the upper- or lowercase letters you typed. If you assign an incorrect value to an entry, possibly mistyping it, simply repeat the command with the correct value. SET will overwrite the existing value. The following command overwrites the current value for the BOOK environment entry with "DCR Fourth Edition":

```
C:\> SET  BOOK=DCR Fourth Edition <ENTER>
```

Using SET, you can display the environment's current contents to see that the former BOOK entry has been overwritten.

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
BOOK=DCR Fourth Edition
```

Environment Space Is Not Unlimited

By default, DOS sets aside space for 160 bytes of environment information (256 bytes in DOS 5 or later). If the number of entries in your environment exceeds this amount, DOS will try to expand the environment. If DOS cannot increase the environment's size, DOS will display the following error message:

```
Out of environment space
```

If this error message occurs, turn to "Increasing the Environment Size During System Startup" in the "Advanced Concepts" part of this chapter.

Removing an Environment Entry

Removing an environment entry is similar to adding an entry—you specify an environment entry name followed by an equal sign. The difference is that you don't specify a corresponding value—you set it equal to "nothing." The command shown in the following example removes the BOOK entry.

```
C:\> SET BOOK= <ENTER>
```

Using SET, you can see the entry is gone:

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
```

If you misspell an entry name or specify a nonexistent entry, SET does not display an error message; SET simply ignores the command.

How DOS Uses the Environment

In addition to the SET command, the PROMPT, PATH, and APPEND commands add entries to the DOS environment. Each time a command completes, for example, DOS searches the environment for a PROMPT= entry. If such an entry exists, DOS uses the entry's value to customize your system prompt. If the entry does not exist, DOS displays its default prompt, which contains only a drive letter followed by a greater-than sign (C >).

When DOS fails to locate a command as an internal command in memory, or as an external command or batch file in the current directory, DOS searches the directories specified in the command path. DOS uses the environment PATH= entry to determine which directories to search.

As you become more conversant with DOS batch files, you can create batch files that access environment entries to perform specific processing.

DOS 5 or later Environment Entries

If you are using DOS 5 or later, DOS provides two special environment entries, TEMP and DIRCMD, that can increase your capabilities. As you will learn in Chapter 21, the DOS pipe operator lets you direct one command's output to become the input of a second command. For example, the following command uses the pipe operator (|) to display a sorted directory listing.

```
C:\> DIR  |  SORT <ENTER>
```

To implement the pipe operator, DOS actually creates temporary files in the root directory of your disk that exist while the command executes. If the current disk has limited available disk space, DOS may not be able to create the temporary files and the command will fail. The TEMP environment entry lets you tell DOS 5 or later where you want it to create these temporary files. For example, the following command assigns TEMP the value of the fast RAM drive E:

```
C:\> SET TEMP=E: <ENTER>
```

Because RAM drives use your computer's electronic memory, they are much faster than your mechanical disk drive. By directing DOS to create its temporary files on the RAM drive, the commands execute quicker and reduce unnecessary wear on your disk. For more information on RAM drives, refer to Chapter 24.

As discussed in Chapters 4 and 18, the DOS 5 or later DIR command supports several additional command-line switches, as shown in Table 19-1.

The following DIR command, for example, displays a directory listing sorted by size, omitting directory names, with the filenames and extensions in the lower case.

```
C:\> DIR  /A:-D  /O:S  /L <ENTER>
```

As you become more comfortable with the DIR command switches, you will eventually select several switches that best serve your needs. As you can imagine, having to type a complex series of switches each time you invoke DIR could become tedious very quickly. The DOS 5 or later DIRCMD environment entry lets you specify the options you want DOS to use by default each time you use DIR. The following SET command selects the settings used in the previous DIR command as the default:

```
C:\> SET  DIRCMD=/A:-D/O:S/L <ENTER>
```

Once you define the DIRCMD entry, you no longer have to repeatedly type the switches. In DOS 5 or later, each time you invoke DIR to display a directory listing, DIR searches for DIRCMD= in the environment entries. If a matching entry exists, DIR uses those switches to format the default directory listing.

Switch	**Function**	
/A:*Attributes*	Displays only those files with the specified attributes:	
	H	Hidden files
	–H	Exclude hidden files
	S	System files
	–S	Exclude system files
	D	Directories
	–D	Exclude directories
	A	Files requiring archiving
	–A	Files successfully archived
	R	All read-only files
	–R	Exclude read-only files
/O:*SortOrder*	Sorts the directory listing by one or more of the following orders:	
	N	Name, from A to Z
	–N	Name, from Z to A
	E	Extension, from A to Z
	–E	Extension, from Z to A
	D	Date and time, oldest to newest
	–D	Date and time, newest to oldest
	S	Size, smallest to largest
	–S	Size, largest to smallest
	G	Group, directories displayed before files
	–G	Group, files displayed before directories
	C	Compression ratio, lowest to highest (DOS 6)
	–C	Compression ratio, highest to lowest (DOS 6)
/S	Displays files in lower-level directories	
/B	Displays only filenames and extensions, one per line	
/L	Displays filenames and extensions in lower case	
/C	Displays compression ratios of DOS 6 DBLSPACE volumes	

Table 19-1. Command-Line Switches Supported by the DOS 5 or later DIR Command

SUMMARY

Displaying Environment Entries

The SET command displays the current environment entries. Environment entries have the form *NAME=Value.* To display the entries, execute the SET command with no parameters:

```
C:\> SET <ENTER>
```

Adding Environment Entries

To add an entry to the DOS environment, invoke SET with the environment name and value, as shown here:

```
C:\> SET EntryName=DesiredValue <ENTER>
```

SET will convert the entry name to uppercase letters, leaving the entry's value unchanged.

Removing Environment Entries

To remove an existing environment entry, invoke the SET command with the entry name followed by an equal sign but no value to the right of the equal sign.

Hands On

Using the SET command, display your current environment entries.

```
C:\> SET <ENTER>
```

Next, create an entry called CHAPTER and assign it the value 19.

```
C:\> SET CHAPTER=19 <ENTER>
```

Use SET to display the environment's new contents. As discussed, to change an entry's value, simply invoke SET a second time specifying the desired value. In this case, change the CHAPTER entry to "Nineteen."

```
C:\> SET CHAPTER=Nineteen <ENTER>
```

Use SET to display the entry's new value, and then remove the entry.

```
C:\> SET CHAPTER= <ENTER>
```

If you are using DOS 5 or later, direct DIR to display filenames and extensions only, using lowercase letters, by setting the DIRCMD entry as follows:

```
C:\> SET  DIRCMD=/B/L <ENTER>
```

Experiment with different DIRCMD settings, invoking DIR each time until you determine the setting you prefer. Next, consider placing the corresponding SET DIRCMD command in your AUTOEXEC.BAT file so the setting is in place each time your system starts.

Review

1. What is the DOS environment?
2. Which DOS commands place entries in the environment?
3. How do the following commands differ?

   ```
   SET BOOK=DOS The Complete Reference
   SET book=DOS The Complete Reference
   ```

4. How do you change an existing environment entry's value?
5. What is the purpose of the DOS 5 or later TEMP and DIRCMD environment entries?

Answers

1. Each time DOS starts, it allocates a portion of memory for the storage of system variables such as the command path and current system prompt. This area of memory is the DOS environment. Using the SET command, you can display, add, or remove environment entries.
2. The SET, APPEND, PROMPT, and PATH commands may add or update environment entries.
3. These two commands are the same. The SET command always converts an entry's name, in this case BOOK, to uppercase.
4. You change an existing environment entry's value by invoking SET with the entry name and desired value, as if you were setting the value for the first time.
5. The TEMP environment entry tells DOS where you want it to create its temporary files. The DIRCMD entry tells DIR the switches you want it to use by default. These switches let you specify the directory list sort order, whether the listing is in upper- or lowercase letters, and even which file types DIR displays.

Advanced Concepts

The "Advanced Concepts" section of Chapter 17 discussed how to access DOS environment entries from within your batch file using named parameters. To use named parameters, you must be using DOS 3.3 or later. In general, a named parameter is simply the name of an environment entry surrounded by percent signs, for which a value is returned. For example, the following batch file displays the current value of the PATH entry:

```
@ECHO OFF
ECHO %PATH%
```

When DOS encounters letters enclosed by percent signs as it executes a batch file, DOS searches the environment for a matching entry, substituting the entry's value for the name. If the entry does not exist, DOS replaces the named parameter with the null string. The following batch file tests whether the named parameter BOOK is defined (not equal to the null string) and, if it is, displays BOOK's value. If the BOOK is undefined, the batch file displays a message saying so.

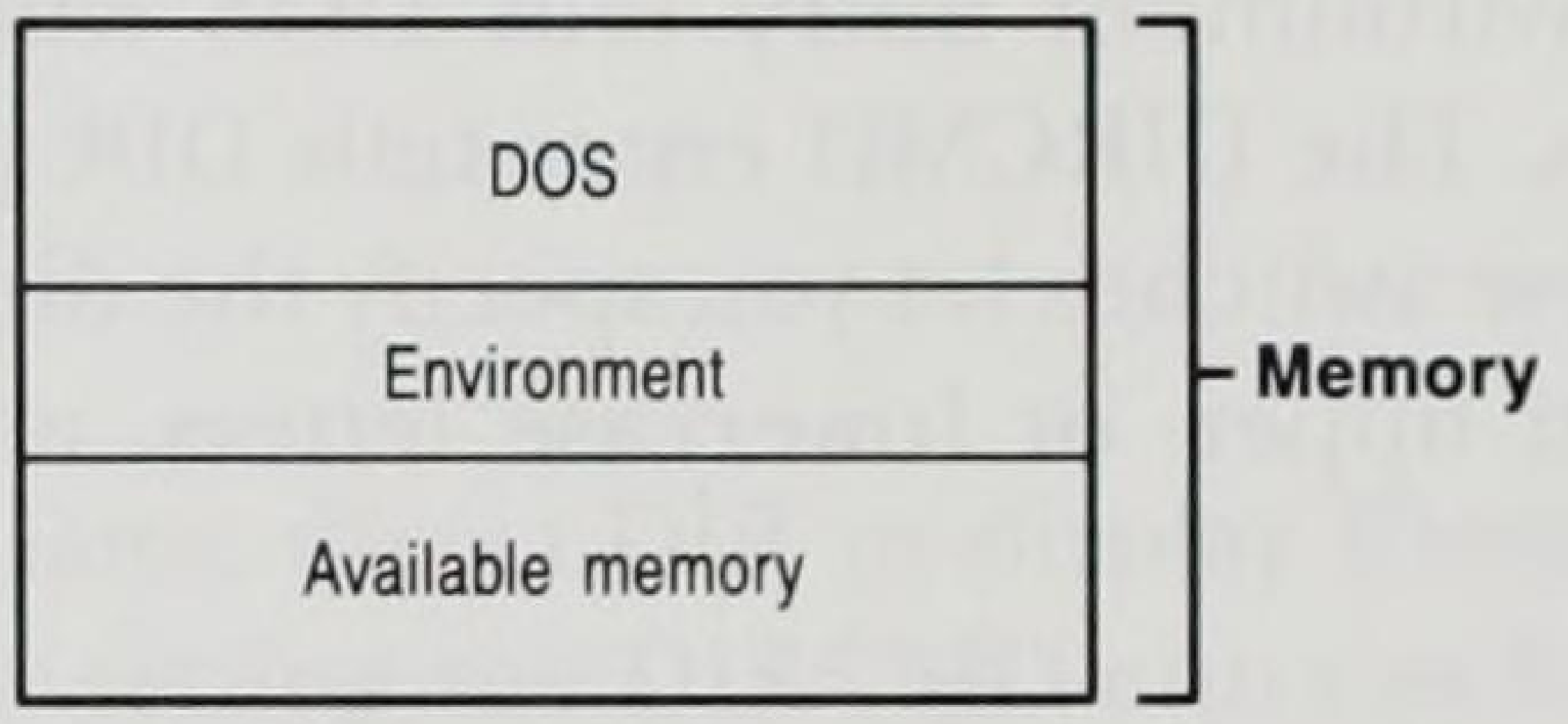
```
@ECHO OFF
IF '%BOOK%'=='' ECHO BOOK is undefined
IF NOT '%BOOK%'=='' ECHO BOOK's value is %BOOK%
```

Increasing the Environment Size During System Startup

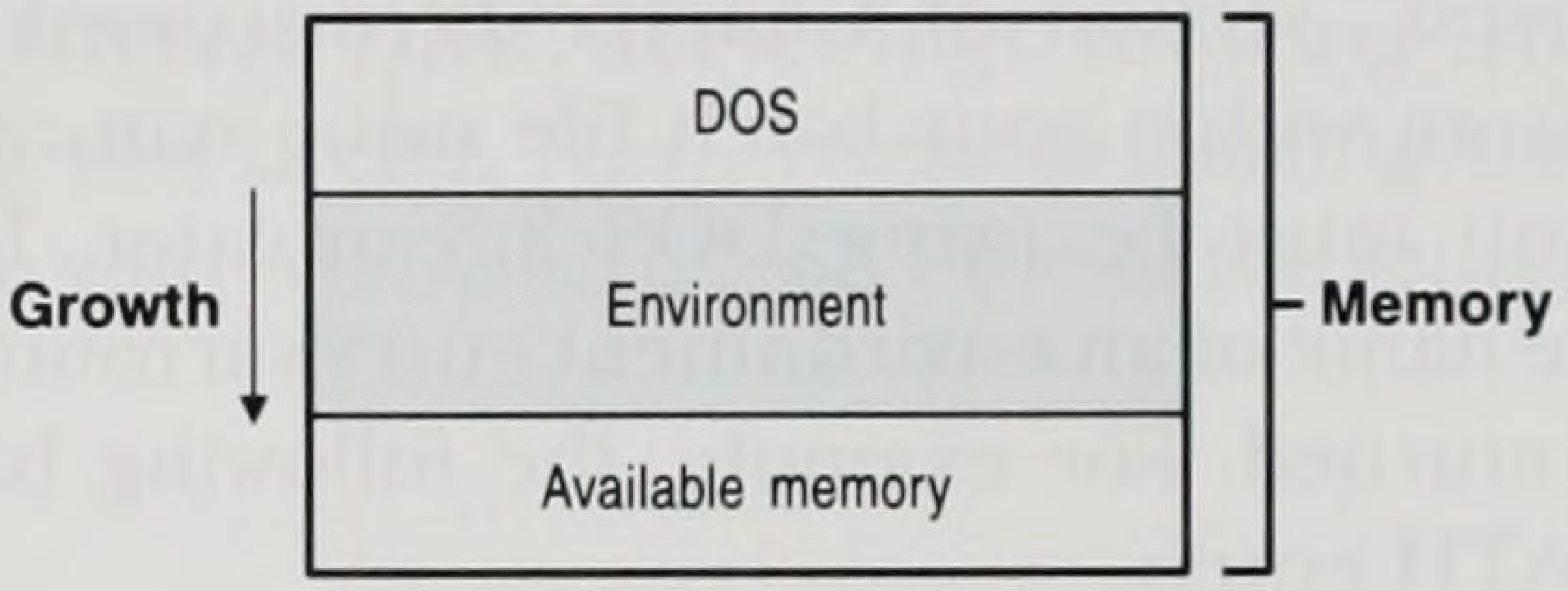
By default, DOS allocates either 160 or 256 bytes for the environment, depending on which version you use. If the total number of characters in all your environment entries exceeds the allocated amount, DOS attempts to increase the environment's size to accommodate them. If DOS has room to increase the environment's size, it does. If DOS cannot increase the environment size, the SET command fails, and DOS displays the following error message:

```
Out of environment space
```

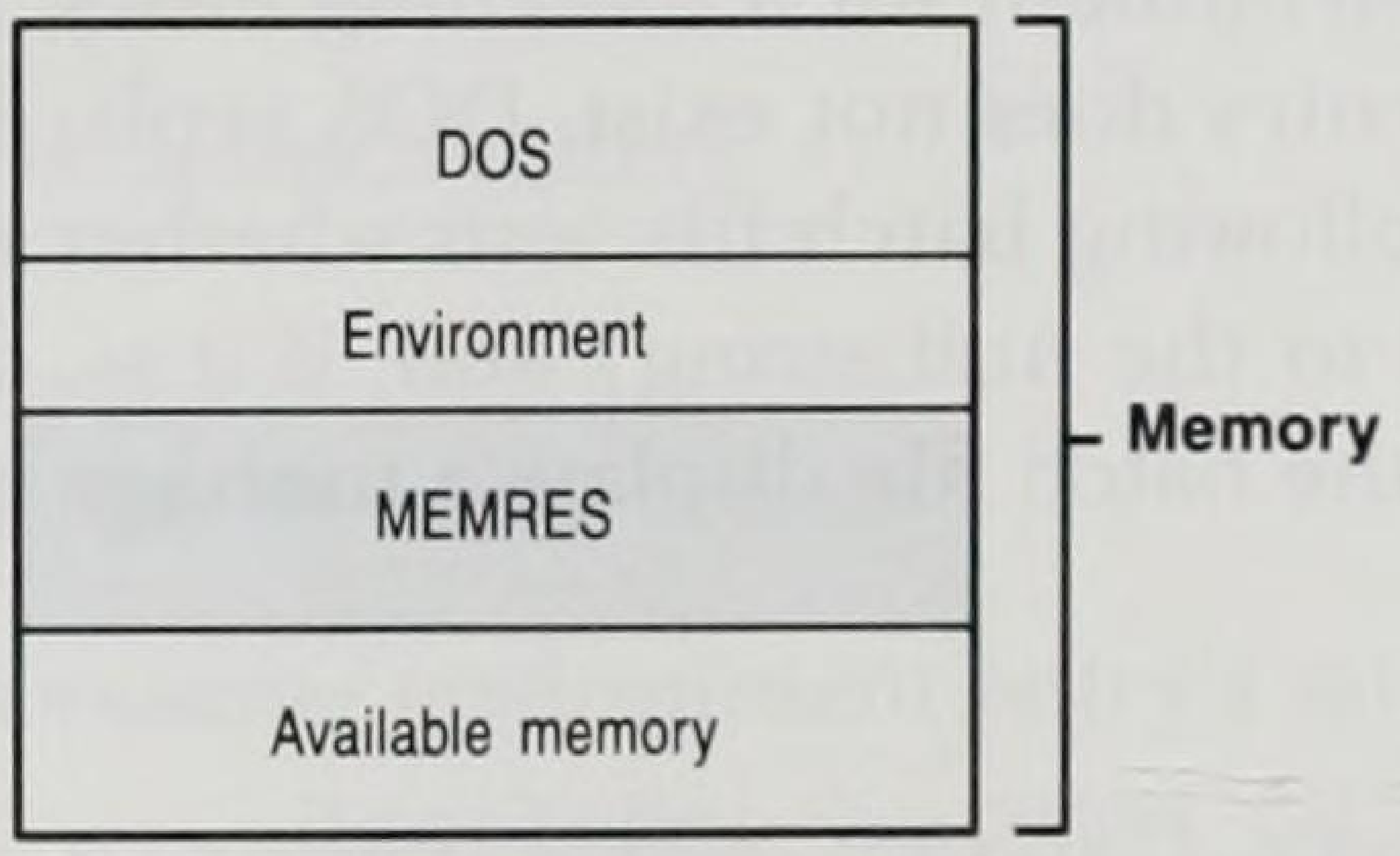
To better understand why there are times when the environment can grow and times when it can't, you need to examine a simplified DOS memory map, shown here:


DOS
Environment
Available memory
Memory

As you can see, the DOS environment resides in an area of memory next to the memory available for use. When the size of the environment exceeds the allocated space, DOS can increase the environment's size, as shown in the following:


DOS
Growth
Environment
Available memory
Memory

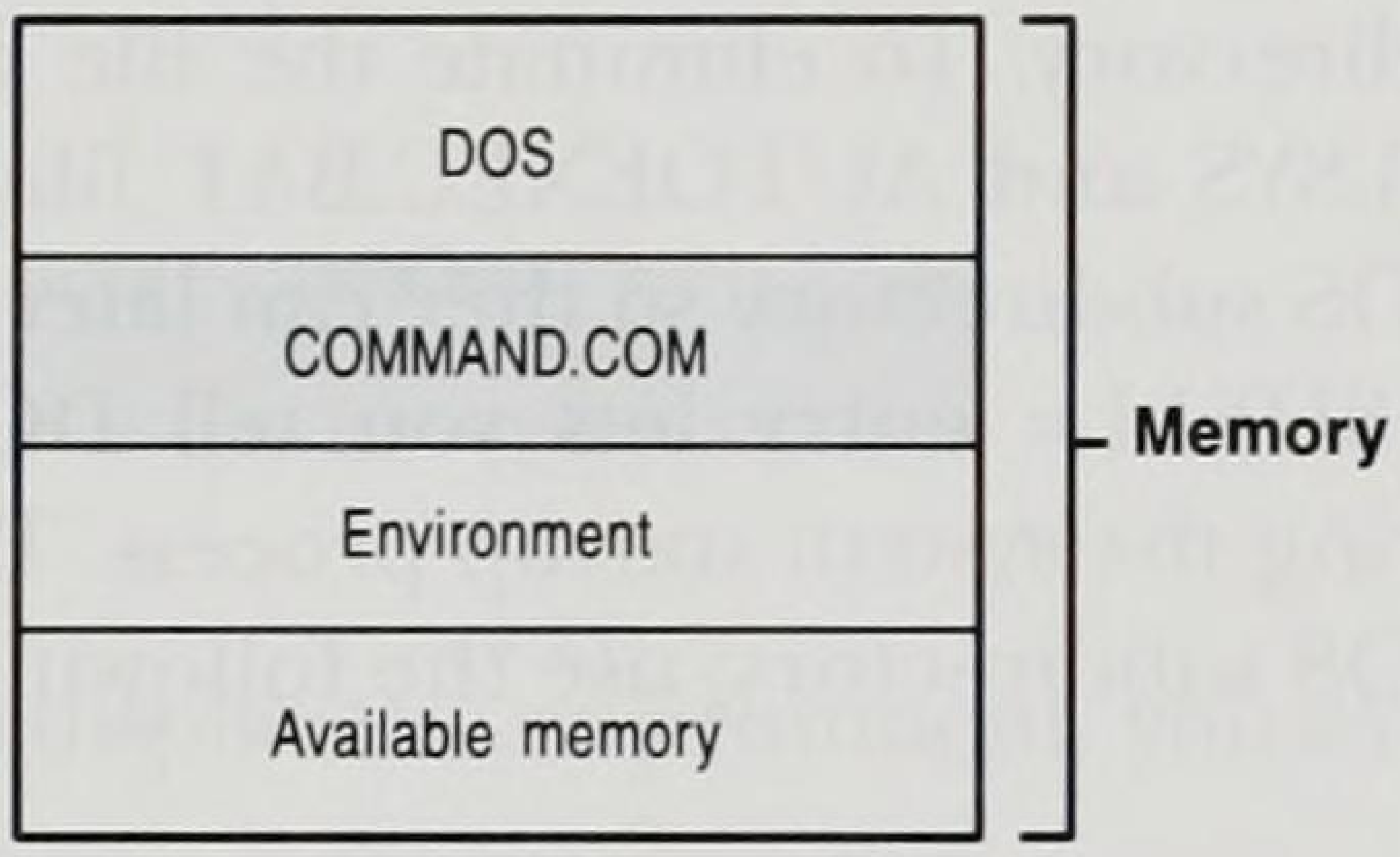
When you install memory-resident software, such as GRAPHICS, PRINT, or FASTOPEN, DOS uses a portion of the available memory, as shown here:


DOS
Environment
MEMRES
Available memory
Memory

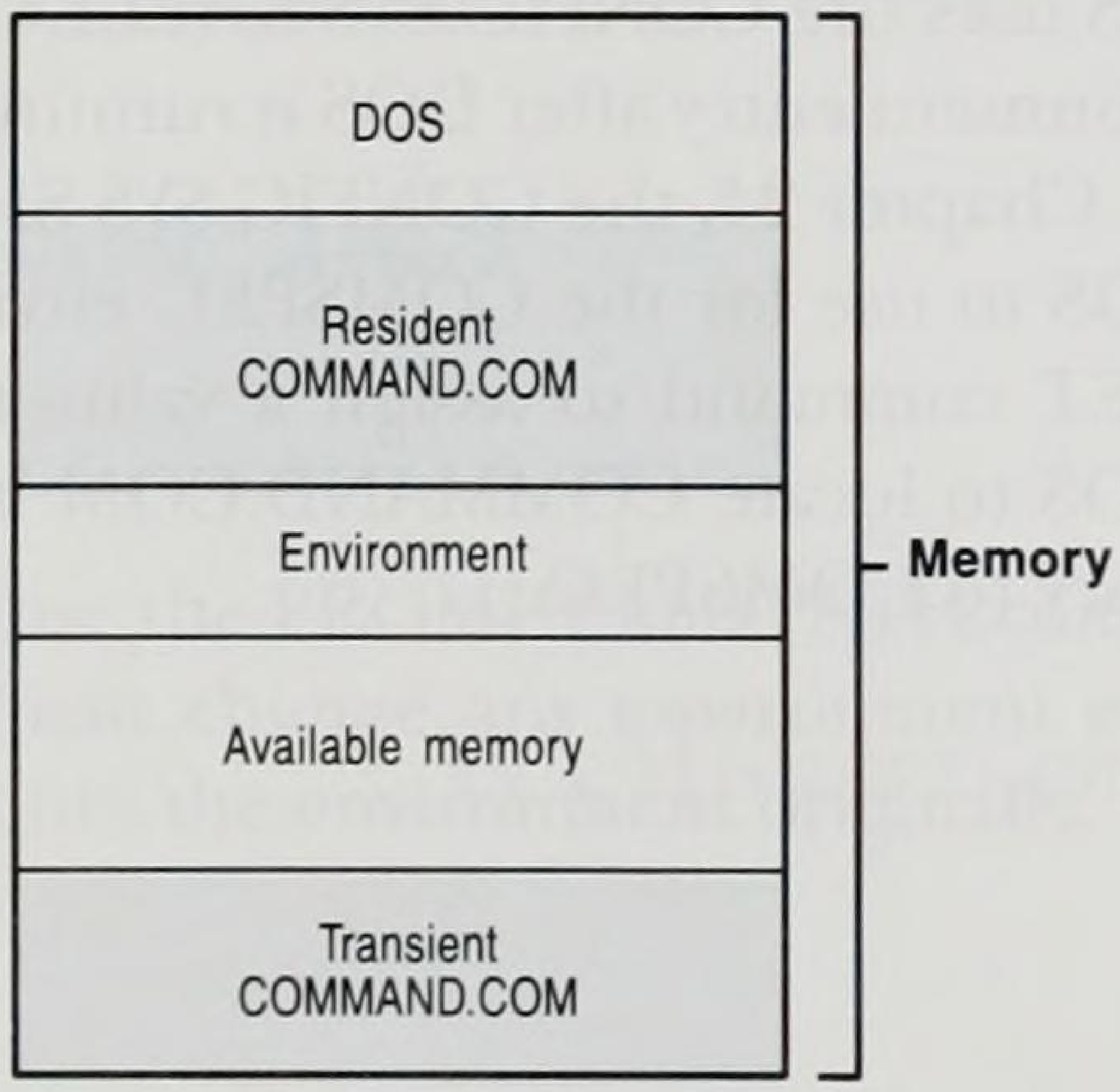
If DOS tries to increase the environment's size now, DOS fails because the memory-resident program is there.

If you experience the "Out of environment space" error message, you may be able to correct the problem by restarting DOS and issuing all of your SET commands before loading any memory-resident programs. If this technique fails, you can direct DOS to reserve more memory for the environment each time DOS starts by using the CONFIG.SYS SHELL= entry mentioned in Chapter 18 and discussed in detail in Chapter 24.

One of the entries DOS always places in the environment is COMSPEC=, which tells DOS where to locate the command processor once the system is running. COMMAND.COM contains the software responsible for displaying the DOS prompt and executing the commands you enter. In addition, COMMAND.COM contains all the internal DOS commands. Each time your system starts, DOS loads the command processor into memory, as shown here:



To make better use of your computer's memory, DOS divides the command processor into two parts—a *resident part,* which is always in memory, and a *transient part,* whose contents DOS can overwrite when it needs to execute a large program. The following illustration shows both the resident and transient portions of COMMAND.COM.



In the cases when DOS overwrites the transient portion of the command processor when a program executes, DOS must reload the command processor when the program completes. To do this, DOS reloads a portion of COMMAND.COM. The COMSPEC= environment entry tells DOS where to find COMMAND.COM. If DOS cannot locate COMMAND.COM, your system will hang and you must restart DOS. By default, COMSPEC= contains the drive letter and root directory of the boot disk, as shown here:

```
C:\> SET
COMSPEC=C:\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
```

Most users have a copy of the file COMMAND.COM in their disk's DOS subdirectory and in the root directory. To eliminate the file duplication, many users will change their CONFIG.SYS and AUTOEXEC.BAT files to use the copy of COMMAND.COM in the DOS subdirectory so they can later delete the copy in the root.

The CONFIG.SYS SHELL= entry lets you tell DOS where to locate the file COMMAND.COM during the system startup process. To start DOS with the copy of CONFIG.SYS in the DOS subdirectory, use the following CONFIG.SYS entry:

```
SHELL=C:\DOS\COMMAND.COM /P
```

Should DOS later need to read the transient portion of COMMAND.COM back into memory, it locates COMMAND.COM using the COMSPEC= environment entry. Therefore, you must place a SET command in your AUTOEXEC.BAT that tells DOS to locate COMMAND.COM in the DOS subdirectory, for example:

```
SET COMSPEC=C:\DOS\COMMAND.COM
```

To summarize, DOS uses the CONFIG.SYS SHELL= entry during system startup and COMSPEC= environment entry after DOS is running to find COMMAND.COM.

As you will learn in Chapter 24, the CONFIG.SYS SHELL= entry lets you specify the value you want DOS to use for the COMSPEC environment entry, eliminating your need to use a SET command to assign a value to the entry. The following SHELL= entry tells DOS to locate COMMAND.COM in the DOS directory and to assign the DOS directory to COMSPEC:

```
SHELL=C:\DOS\COMMAND.COM C:\DOS /P
```

In addition to letting you move and use COMMAND.COM, the CONFIG.SYS SHELL= entry also lets you define the desired environment size. To specify an environment size, simply include with the SHELL= entry the /E switch, specifying the number of bytes desired, from 160 to 32,768. The following entry reserves an environment of 512 bytes.

```
SHELL=C:\DOS\COMMAND.COM /P /E:512
```

Using SET to Change Your Command Path or System Prompt

As you know, the PROMPT command defines your system prompt, and the PATH command defines your command search path. Both the PROMPT and PATH commands create environment entries. Since the SET command can change or remove any environment entry, you can change your system prompt or command path using SET. For example, the following command uses SET to change the current prompt to "What now?".

```
C:\> SET PROMPT=What now? <ENTER>
What now?
```

Using the SET command to display the environment, you will see the new value for the PROMPT entry.

```
What now? SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=What now?
```

To restore the prompt, you can use either the PROMPT or SET command.

```
What now?  SET PROMPT=$P$G <ENTER>
C:\>
```

Although you will normally use the PROMPT and PATH commands instead of SET, you should know that SET can change any environment entry, no matter which command placed the entry into the environment originally.

Key Terms

DOS environment An area of memory DOS reserves during system startup for the storage of such system variables as the current system prompt or command path.

Environment entry A name and value combination stored in the DOS environment that takes the form *NAME=Value.*

Transient command processor The portion of the DOS command processor (COMMAND.COM) that DOS periodically overwrites during the execution of a large program. When this occurs, DOS must reload the software from disk.

Resident command processor The portion of the DOS command processor that always remains resident in your computer's memory while DOS is active. A major task of the resident portion of the command processor is reloading the transient portion as necessary.

Chapter 20

The DOS Redirection Operators

So far (except in Chapter 13), each time you have executed a command, DOS has displayed the command's output to your screen. For most of your applications, displaying the output on the screen makes perfect sense. For some DOS commands, such as DIR, there may be times you would rather have DOS send the program's output elsewhere, perhaps to your printer or to a file. In this chapter you will examine the DOS output and append *redirection operators.* These operators are so named because they instruct DOS to redirect the output destination from the screen.

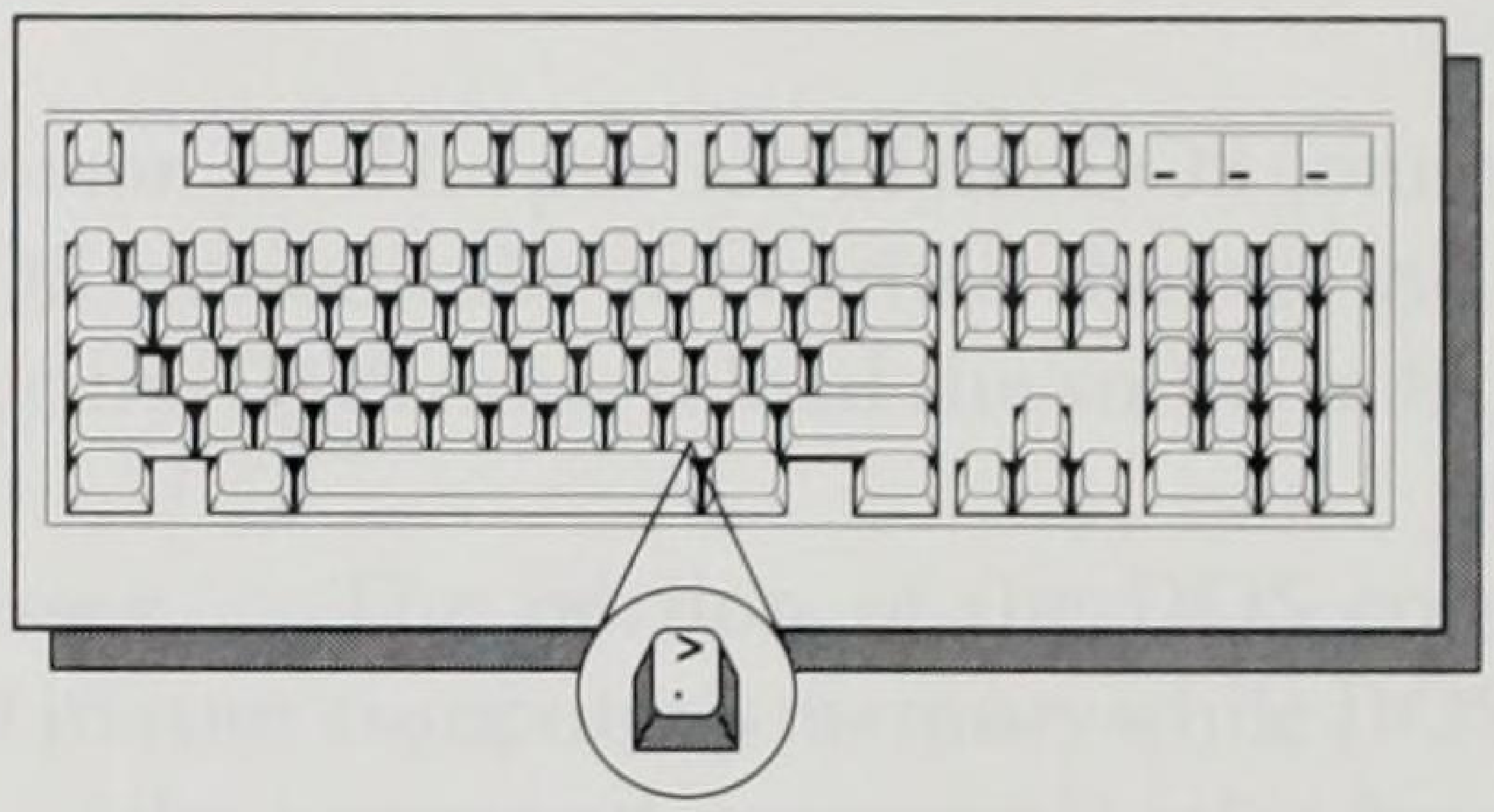
In addition to the output redirection operators, DOS provides an input redirection operator that lets a command get its input from a file, as opposed to the keyboard. The discussion of the DOS input redirection operator presents three new DOS commands: MORE, SORT, and FIND.

The Output Redirection Operator

By default, all DOS commands display their output to your screen. In some cases, however, you may find it more convenient for DOS to direct a command's output to the printer or to a file. The command in the following example sends the output of the DIR command to your printer. Printing a directory listing in this manner provides an effective way to catalog the files on a disk.

```
C:\> DIR > PRN <ENTER>
```

The greater-than symbol (>) is the *output redirection operator.* The following illustration shows the location of the output redirection key on your keyboard. You must press your keyboard's right or left SHIFT key to access the greater-than character.



When it encounters the output redirection operator in a command line, DOS sends the output of the command specified on the left of the operator to the file or device specified on the right of the operator. In the previous example, DIR is the command whose output DOS redirects from the screen to PRN. DOS will print the directory listing in this case and will not display any of the directory listing on the computer screen.

Pictorially, you can visualize the output redirection process as shown in Figure 20-1.

Using the printer once again as the destination for redirected output, the following command prints the name of every file on your disk, including your disk's directory structure:

```
C:\> TREE C:\ /F > PRN <ENTER>
```

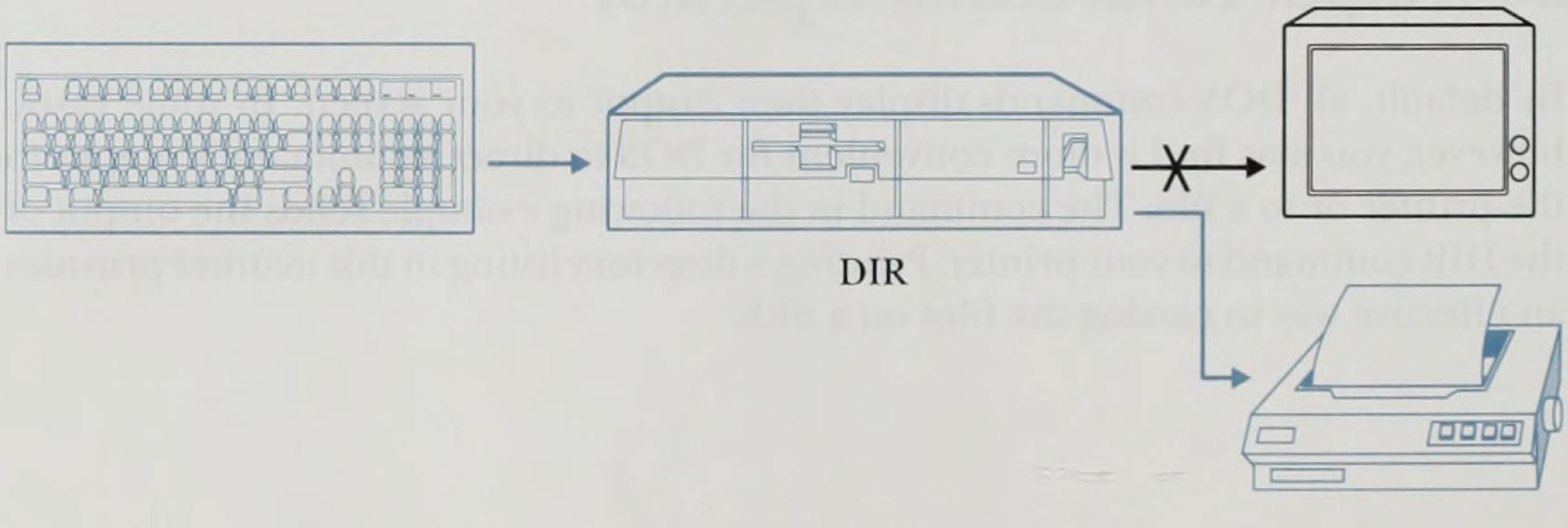


Figure 20-1. Redirecting the output of DIR from the screen to the printer

As you can see, redirecting a program's output to the printer is very convenient when you need a hardcopy listing.

In some cases you might want to redirect a command's output to a file that you can later edit with your word processor or the DOS editor. To redirect a command's output to a file, place the file's name to the right of the redirection operator. The following command line, for example, redirects the output of the TREE command to the disk file TREE.DAT:

```
C:\> TREE  C:\  /F  >  TREE.DAT <ENTER>
```

Pictorially, the redirection from the screen to the file TREE.DAT is shown in Figure 20-2.

The output redirection operator overwrites existing files with the same name as the file specified to the right of the output redirection operator. If you redirect the output of a DOS command to a file and a file with the same name exists, DOS will overwrite the existing file's contents.

WARNING

Not All Commands Support Redirection

There are times when use of the output redirection operator makes obvious sense, such as printing a directory listing or the output of the TREE command. Most applications (your word processor or spreadsheet), however, do not support I /O redirection. If you try to use output redirection with these commands, DOS will either ignore the redirection operator, or the command might fail, possibly causing your system to freeze up. Should you accidentally redirect such a command, first try to use the CTRL-C keyboard combination to cancel the command. If your system remains

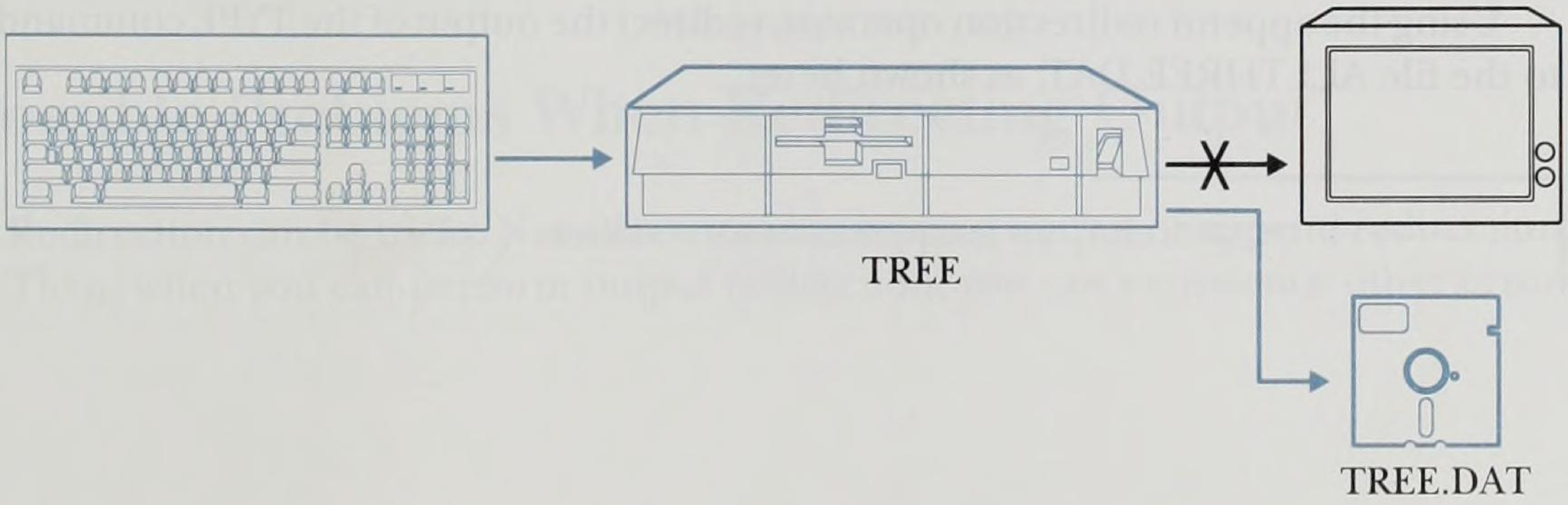


Figure 20-2. Redirecting the output of TREE from the screen to the file TREE.DAT

"hung up," you will need to use the CTRL-ALT-DEL keyboard combination to restart DOS or may even need to turn your computer off and back on.

The Append Redirection Operator

As you have learned, if you use the output redirection operator to route a program's output to a file, DOS will overwrite an existing file. The *append redirection operator* lets you redirect a command's output from the screen appending the output to the end of an existing file. DOS uses two greater-than signs for the append redirection operator (>>).

To better understand how the append redirection operator works, create the files START.DAT, MIDDLE.DAT, and END.DAT using your word processor, the DOS editor, or by copying the files from the keyboard. The first file, START.DAT, will contain

```
UNLV      UNLV      UNLV
```

with six spaces between each entry. The second file, MIDDLE.DAT, will contain

```
RUNNIN    RUNNIN    RUNNIN
```

with four spaces between entries. The last file, END.DAT, will contain

```
REBELS!   REBELS!   REBELS!
```

with three spaces between each.

To begin, use the TYPE command to display the contents of the file START.DAT:

```
C:\> TYPE  START.DAT <ENTER>
UNLV      UNLV      UNLV
```

Using the append redirection operator, redirect the output of the TYPE command to the file ALLTHREE.DAT, as shown here:

```
C:\> TYPE START.DAT >> ALLTHREE.DAT. <ENTER>
```

The append redirection operator attaches the redirected output to the end of an existing file. In this case, because the file ALLTHREE.DAT does not exist, the append redirection creates it.

Again redirecting the TYPE command, append the contents of the file named MIDDLE.DAT to the file ALLTHREE.DAT:

```
C:\> TYPE MIDDLE.DAT >> ALLTHREE.DAT <ENTER>
```

If you display the contents of the file ALLTHREE.DAT, you will find DOS has successfully appended the file's contents.

```
C:\> TYPE  ALLTHREE.DAT <ENTER>
UNLV      UNLV      UNLV
RUNNIN    RUNNIN    RUNNIN
```

Next, the following command appends the last section of information to the file ALLTHREE.DAT:

```
C:\> TYPE END.DAT >> ALLTHREE.DAT <ENTER>
```

Using the TYPE command to display the contents of ALLTHREE.DAT, the file's final contents has become the following:

```
C:\> TYPE  ALLTHREE.DAT <ENTER>
UNLV      UNLV      UNLV
RUNNIN    RUNNIN    RUNNIN
REBELS!   REBELS!   REBELS!
```

Possible Problems When Redirecting Output

Redirection can be tricky. Not all commands support output or append redirection. Then, when you can perform output redirection, you can experience other errors that may be difficult at first to understand. First, if you are redirecting a program's output to your disk, you must ensure that your disk has adequate available space. If your disk runs out of space during a command's execution, DOS will display the following error message:

```
Insufficient disk space
```

Second, as you may remember from Chapter 1, DOS cannot store information on a write-protected disk. If you try to redirect a command's output to a write-protected disk, DOS will display the following error message:

```
Write protect error writing drive A
Abort, Retry, Fail?
```

In Chapter 22 you will learn that DOS lets you protect a file from being accidentally deleted or overwritten by using the ATTRIB command to set the file to read-only. If you try to overwrite a read-only file using output redirection or if you try to append additional information to the file, the command will fail and DOS will display an error message.

Last, a difficult redirection error to discover occurs when you redirect the output of a command that expects keyboard input. In Chapter 4 you learned that the /P switch directs DIR to pause with each screenful of directory entries until you press a key to continue. If you redirect to the printer or a file, a DIR command that uses /P, DOS will also redirect the prompt for you to press a key to continue. As a result, the command won't complete. Instead, DIR will wait for you to respond to a prompt you can't see. Your system will appear to be hung up, and your screen will only display a blinking cursor. Should an error such as this occur, first try to cancel the command using the CTRL-C keyboard combination. If you cannot cancel the command, you will have to restart your system by pressing CTRL-ALT-DEL or by turning your computer off and then back on.

The Input Redirection Operator

You have learned that DOS writes a command's output to the screen display unless you tell it otherwise. In the same way, DOS, by default, gets a command's input from the keyboard. The *input redirection* operator lets you direct DOS to get a command's input from an existing file, as opposed to the keyboard. DOS uses the less-than character (<) as the input redirection operator.

As you will learn, only a few DOS commands support input redirection. In this chapter, we will examine three commands specifically designed for input redirection: MORE, SORT, and FIND.

The MORE Command

The MORE command displays its input a screenful at a time, pausing with each screenful of information, for you to press a key to continue. Assume, for example, that you have a large text file named REPORT.DAT. If you try to display the file's contents using the TYPE command, the information may scroll past you on the screen faster than you can read it. Using the MORE command and input redirection, you can display the file's contents a screenful at a time, as shown here:

```
C:\> MORE  <  REPORT.DAT <ENTER>
```

Each time MORE displays a screenful of information, MORE displays the following prompt giving you time to view the screen's contents.

```
—More—
```

If you want to view the next screenful of information, simply press any key. If you have seen enough, use CTRL-C to cancel the command. You can picture the input redirection to MORE as shown in Figure 20-3.

MORE is an external DOS command residing in the file MORE.COM.

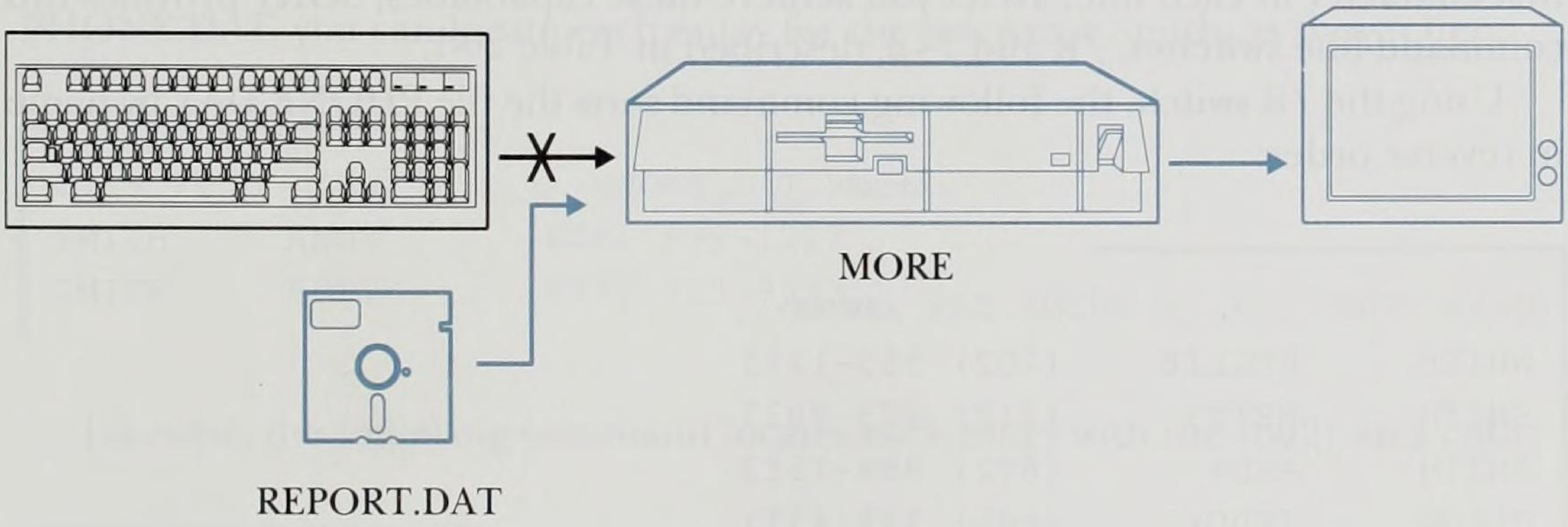


Figure 20-3. Redirecting the input of MORE from the keyboard to the file REPORT.DAT

The SORT Command

The SORT command lets you sort the contents of an existing file. For example, assume you have a file named PHONES.DAT that contains a list of names, addresses, and phone numbers, as shown here:

```
SMITH     ANDY      (602) 999-1212
JONES     ROBERT    (206) 888-3131
DAVIS     WALLY     (303) 332-2222
JAMSA     STEPHANIE (702) 555-1212
SMITH     BETTY     (212) 223-9933
OLSON     TEDDY     (602) 331-4311
WHITE     KELLIE    (702) 555-1213
```

Using input redirection and the SORT command, you can sort the file's contents by name, for example:

```
C:\> SORT  <  PHONE.DAT <ENTER>
DAVIS     WALLY     (303) 332-2222
JAMSA     STEPHANIE (702) 555-1212
JONES     ROBERT    (206) 888-3131
OLSON     TEDDY     (602) 331-4311
SMITH     ANDY      (602) 999-1212
SMITH     BETTY     (212) 223-9933
WHITE     KELLIE    (702) 555-1213
```

By default, the SORT command sorts its input from lowest to highest based on the first letter of each line. You might, however, want to sort the file in descending order (highest to lowest), or sort the input on a character position other than the first character in each line. To let you achieve these capabilities, SORT provides two command-line switches, /R and /+*n*, described in Table 20-1.

Using the /R switch, the following command sorts the file PHONE.DAT by name in reverse order:

```
C:\> SORT  /R  < PHONE.DAT <ENTER>
WHITE     KELLIE    (702) 555-1213
SMITH     BETTY     (212) 223-9933
SMITH     ANDY      (602) 999-1212
OLSON     TEDDY     (602) 331-4311
JONES     ROBERT    (206) 888-3131
JAMSA     STEPHANIE (702) 555-1212
DAVIS     WALLY     (303) 332-2222
```

Switch	Function
/R	Performs a descending (reverse) order sort
/+*n*	Sorts the input based on column *n* (a number)

Table 20-1. *SORT's Command-Line Switches*

Likewise, assuming the phone numbers begin at column 22, the following command sorts the entries by phone number:

```
C:\> SORT  /+22  <  PHONE.DAT <ENTER>
JONES     ROBERT    (206) 888-3131
SMITH     BETTY     (212) 223-9933
DAVIS     WALLY     (303) 332-2222
OLSON     TEDDY     (602) 331-4311
SMITH     ANDY      (602) 999-1212
JAMSA     STEPHANIE (702) 555-1212
WHITE     KELLIE    (702) 555-1213
```

The SORT command is an external command residing in the file SORT.EXE.

The FIND Command

The FIND command lets you search redirected input, displaying only those lines containing a specific word or phrase. In other words, the command "finds" and displays each line that contains a specific word or sequence of words. Using the file PHONE.DAT, you can locate each entry for the last name Smith, as shown here:

```
C:\> FIND  "SMITH"  <  PHONE.DAT <ENTER>
SMITH     ANDY      (602) 999-1212
SMITH     BETTY     (212) 223-9933
```

Likewise, the following command locates each entry with the (602) area code:

```
C:\> FIND  "(602)"  <  PHONE.DAT <ENTER>
SMITH     ANDY      (602) 999-1212
OLSON     TEDDY     (602) 331-4311
```

To increase your capabilities, FIND, like SORT, provides command-line switches described in Table 20-2.

The following command uses the /C switch to display a count of the number of entries in the file with the (206) area code:

```
C:\> FIND  /C  "(206)"  <  PHONE.DAT <ENTER>
1
```

In a similar way, the next command uses the /N switch to precede each line containing the name SMITH with its corresponding line numbers:

```
C:\> FIND  /N  "SMITH"  <  PHONE.DAT <ENTER>
[1]SMITH      ANDY      (602) 999-1212
[5]SMITH      BETTY     (212) 223-9933
```

The FIND command is an external command residing in the file FIND.EXE. In Chapter 21 you will use FIND in conjunction with the DOS pipe operator.

Combining Input and Output Operators

As you become more comfortable with DOS I/O redirection operators, you may use both operators within the same command line. The following command sorts the file PHONE.DAT and writes the output to the file SORTED.DAT:

```
C:\> SORT  <  PHONE.DAT  >  SORTED.DAT
```

In this case, SORT gets its input from the file PHONE.DAT and DOS redirects SORT's output to the file SORTED.DAT. Picture the redirection as shown in Figure 20-4.

Switch	Function
/C	Displays only a count of the number of lines containing the word or phrase
/I	Considers uppercase and lowercase letters equivalent (DOS 5 or later)
/N	Precedes each matching line with its corresponding line number
/V	Displays the lines that *do not* contain the word or phrase

Table 20-2. FIND's Command-Line Switches

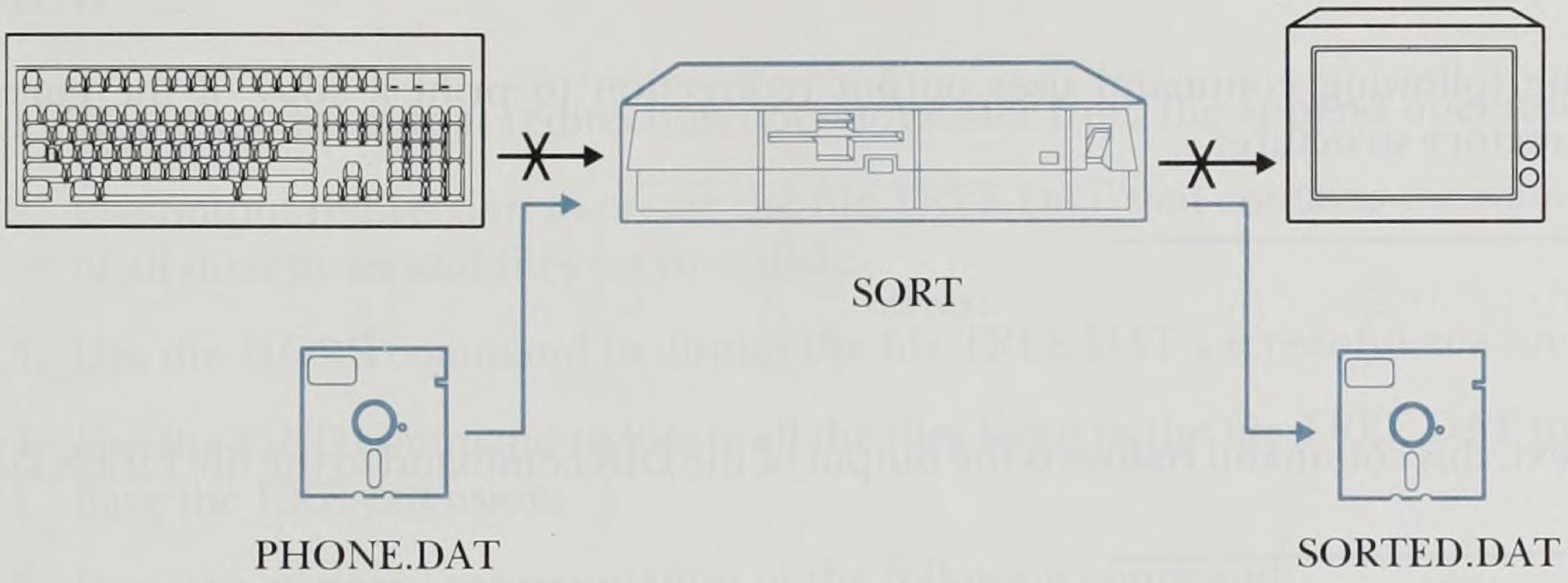


Figure 20-4. Combining the input and output redirection operators

SUMMARY

The Output Redirection Operator (>)

The output redirection operator (>) directs DOS to route a command's output from the screen display to a specified file or device. Most DOS commands support output redirection. If DOS redirects output to a file, it will overwrite a file with the same name. The following DOS command redirects the output of DIR from the screen to a printer.

```
C:\> DIR  >  PRN <ENTER>
```

The Append Redirection Operator (>>)

The append redirection operator directs DOS to route a command's output from the screen display to append the output to the end of an existing file. If the destination file specified on the right of the append redirection operator does not exist, DOS will create the file.

The Input Redirection Operator (<)

The input redirection operator directs DOS to get a command's input from an existing file, as opposed to the keyboard. DOS provides three commands that specifically support input redirection: SORT, which sorts the input; MORE, which displays it a screenful at a time; and FIND, which locates and displays lines with specified strings.

Hands On

The following command uses output redirection to print a copy of the current directory structure:

```
C:\> TREE  C:\  >  PRN <ENTER>
```

Next, this command redirects the output of the DIR command to the file FILES.DAT:

```
C:\> DIR  >  FILES.DAT <ENTER>
```

Using the file FILES.DAT and the input redirection operator, the following command displays a sorted directory listing:

```
C:\> SORT  <  FILES.DAT <ENTER>
```

Using SORT's /R switch, sort the file in reverse order:

```
C:\> SORT  /R  <  FILES.DATE <ENTER>
```

Likewise, the following command uses FIND to locate and display only those lines containing the word <DIR>—in other words, all of the subdirectory names:

```
C:\> FIND  "<DIR>"  <  FILES.DAT <ENTER>
```

Using FIND's /C switch, display the number of lines containing the word <DIR>:

```
C:\> FIND  /C  "<DIR>"  <  FILES.DAT <ENTER>
```

Last, using FIND's /V switch, display the lines that don't contain <DIR>:

```
C:\> FIND  /V  "<DIR>"  <  FILES.DAT <ENTER>
```

This gives you the capacity in a batch file, for example, to find all the files in a directory that are not SYS and deal with them accordingly.

Review

1. How does the output redirection operator differ from the append operator?
2. Use output redirection to create the file TREE.DAT that contains the names of all directories and files on your disk.
3. Use the MORE command to display the file TREE.DAT a screenful at a time.
4. Use the FIND command to locate all the files listed in the file TREE.DAT that have the EXE extension.
5. Draw the pictorial representation of the following command:

```
C:\> FIND ".COM" < TREE.DAT > PRN
```

Answers

1. The output and append redirection operators both redirect a command's output from the screen to a file or device. If you redirect a command's output to a file and a file exists with the same name, the output redirection operator will overwrite the existing file's contents. The append redirection operator, on the other hand, will append the command's output to the file. If there is no existing file with the same name as the destination file, both the output and append redirection operators will create the file.
2. The output redirection command line to create the file TREE.DAT is

```
C:\> TREE C:\ /F > TREE.DAT <ENTER>
```

3. The MORE command to display the file TREE.DAT a screenful at a time is

```
C:\> MORE < TREE.DAT <ENTER>
```

4. The FIND command to locate all the files listed in the file TREE.DAT with the EXE extension is

```
C:\> FIND ".EXE" < TREE.DAT <ENTER>
```

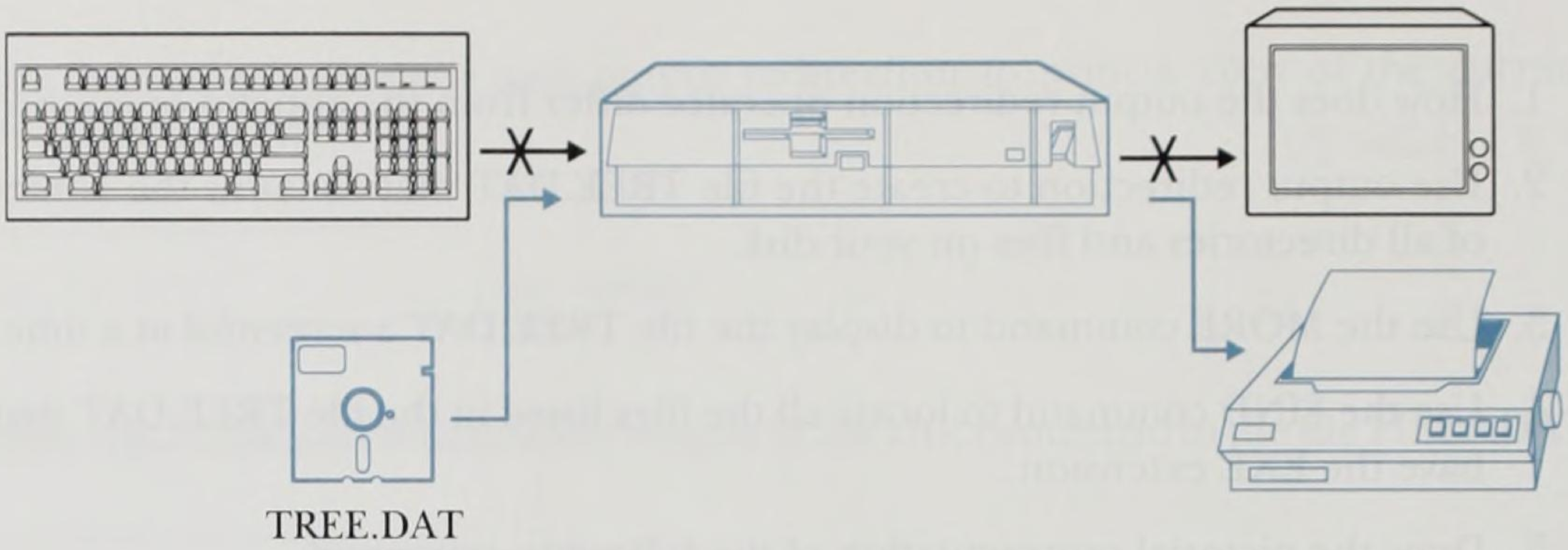


Figure 20-5. *Results of the command C:\> FIND ".COM" < TREE.DAT > PRN*

5. The pictorial representation of the command is shown in Figure 20-5.

Advanced Concepts

Many of the books and articles you will encounter that discuss the I/O redirection operators use the terms *standard input* and *standard output,* or their abbreviations *stdin* and *stdout.* Each time DOS executes a program, DOS assigns default devices for the program. By default, the keyboard is the program's source of input, or standard input device. Likewise, the screen display is the standard output device, as shown in Figure 20-6.

When you use the redirection operators, DOS changes the devices associated with stdin or stdout.

In addition to stdin and stdout, DOS also assigns default devices for a program's printer (stdprn), destination for error messages (stderr), and auxiliary communications (stdaux). Table 20-3 defines these standard devices.

Understanding that DOS assigns these devices each time a program runs is important because these devices affect the number of files a program can open at one time. Each of these devices requires a *file handle.* By default, DOS provides eight file handles to each program. Because these devices use up five of the handles, DOS only leaves your programs with three handles. Many applications (word processor, spreadsheet, or database) need to open several temporary files while they execute. If you do not provide enough file handles for the program to open its files, the program will fail.

The CONFIG.SYS file FILES= entry lets you specify the number of file handles DOS should provide for each program. At a minimum, you should set the value to 20.

```
FILES=20
```

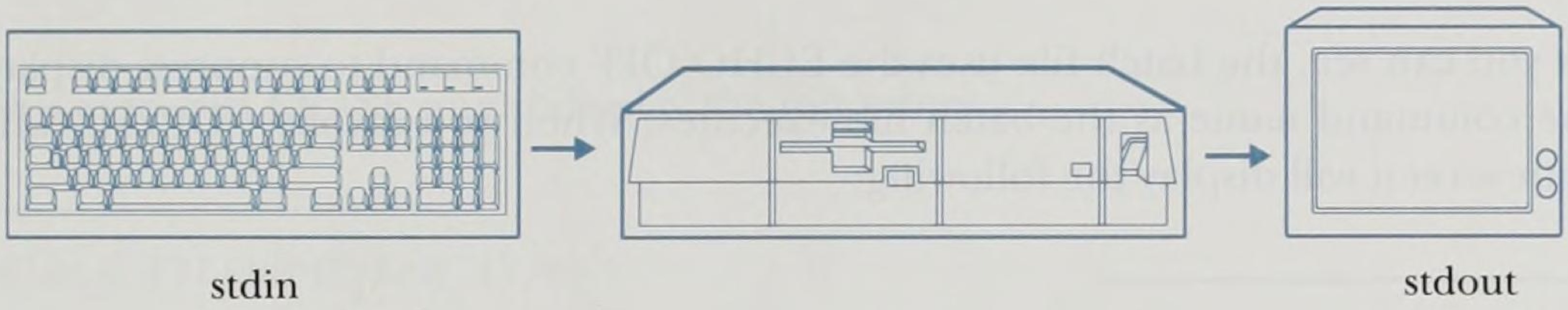


Figure 20-6. *Standard input and output devices*

Knowing that DOS will automatically use the first five handles, your program still has 15 handles available.

Chapter 2 discussed the DOS startup process in detail. At that time you learned DOS uses an initialization program named SYSINIT to perform much of its startup processing. SYSINIT processes the entries in the CONFIG.SYS file and then invokes COMMAND.COM, which in turn either invokes AUTOEXEC.BAT or DATE and TIME. Before SYSINIT invokes COMMAND.COM, however, SYSINIT opens the files stdin, stdout, stderr, stdaux, and stdprn. Many applications will specify the minimum number of file handles they require. Now you should be better able to understand why.

Redirecting Output to the NUL Device

In addition to the device names PRN, AUX, and LPT1, DOS also defines the device NUL for the nonexistent device. Using NUL, your batch files can discard output from commands that would otherwise clutter the screen. For example, the following batch file COPYMSG.BAT copies the contents of your AUTOEXEC.BAT and CONFIG.SYS files to files with the SAV extension:

```
@ECHO OFF
ECHO Backing up AUTOEXEC.BAT and CONFIG.SYS
```

Device Name	Function	Default Device
stdin	Standard input	Keyboard
stdout	Standard output	Screen
stderr	Standard error	Screen
stdprn	Standard printer	LPT1
stdaux	Standard auxiliary	COM1

Table 20-3. *DOS Standard Device Names*

```
COPY  \AUTOEXEC.BAT  \*.SAV
COPY  \CONFIG.SYS  \*.SAV
```

As you can see, the batch file uses the ECHO OFF command to suppress display of the command name as the batch file executes. When you invoke COPYMSG.BAT, your screen will display the following:

```
C:\> COPYMSG <ENTER>
Backing up AUTOEXEC.BAT and CONFIG.SYS
        1 File(s) copied
        1 File(s) copied
```

Although the ECHO OFF command suppresses the display of command names, it does not eliminate the "File(s) copied" messages. However, the following batch file, NOMSG.BAT, discards the messages to the NUL device.

```
@ECHO OFF
ECHO Backing up AUTOEXEC.BAT and CONFIG.SYS
COPY  \AUTOEXEC.BAT  \*.SAV  >  NUL
COPY  \CONFIG.SYS  \*.SAV  > NUL
```

When you invoke this batch file, your screen displays the following:

```
C:\> NOMSG <ENTER>
Backing up AUTOEXEC.BAT and CONFIG.SYS
```

As you can see, the "File(s) copied" messages do not appear. As the complexity of your batch files increases, redirecting output to the NUL device can clean up your screen appearance considerably.

Building Batch Files on the Fly

As you work with DOS, you will eventually begin to build a library of batch files that you use on a regular basis. Depending on your application, there may be times when your batch file needs to invoke another, smaller batch file. For example, the following batch file SELERASE.BAT provides you with selective delete capabilities similar to the DOS 5 DEL /P command. When you invoke SELERASE.BAT with a list of filenames or wildcards, SELERASE will prompt you individually for each file specified to determine if you want to delete that file. For example, the following command directs SELERASE to selectively erase all of the files in the current directory that have the BAK extension:

```
C:\> SELERASE  *.BAK <ENTER>
```

In this case, SELERASE might display the following:

```
Delete FILENAME.BAK (Y/N)?
```

The following batch file implements SELERASE.BAT:

```
@ECHO OFF

REM Loop through the desired file names
:LOOP
  FOR %%I IN (%1) DO CALL DELETEIT %%I
  SHIFT
IF NOT '%1'=='' GOTO LOOP
```

The batch file uses the FOR and SHIFT commands to loop through all of the files specified in the command line. Using FOR, the batch file can support wildcards. The batch file uses a second batch file named DELETEIT.BAT that actually prompts you for whether or not you want to delete the file, and if so, deletes it. The batch file DELETEIT.BAT contains the following:

```
ECHO Delete %1 (Y/N)?
Y_OR_N
IF ERRORLEVEL 1 DEL %1
```

As you can see, the batch file uses the Y_OR_N command that you created in Chapter 16. You may recall that Y_OR_N waits for you to type either **Y** or **N**. If you type **Y**, the command exits with an error level of 1. If you type **N**, the command exits with an error level 0.

Although SELERASE.BAT is very useful, its use of the second batch file, DELETEIT.BAT, wastes disk space. Remember, every time DOS creates a file on your disk, DOS must allocate a cluster of disk space. In the case of DELETEIT.BAT, most of the cluster is unused. A better solution, therefore, is to let SELERASE.BAT create the batch file DELETEIT.BAT "on the fly" each time it executes and then delete the batch file before it completes.

Using the output and append redirection operators with the ECHO command, you can modify SELERASE.BAT to do just that, like this:

```
@ECHO OFF

REM Build the batch file DELETEIT.BAT
```

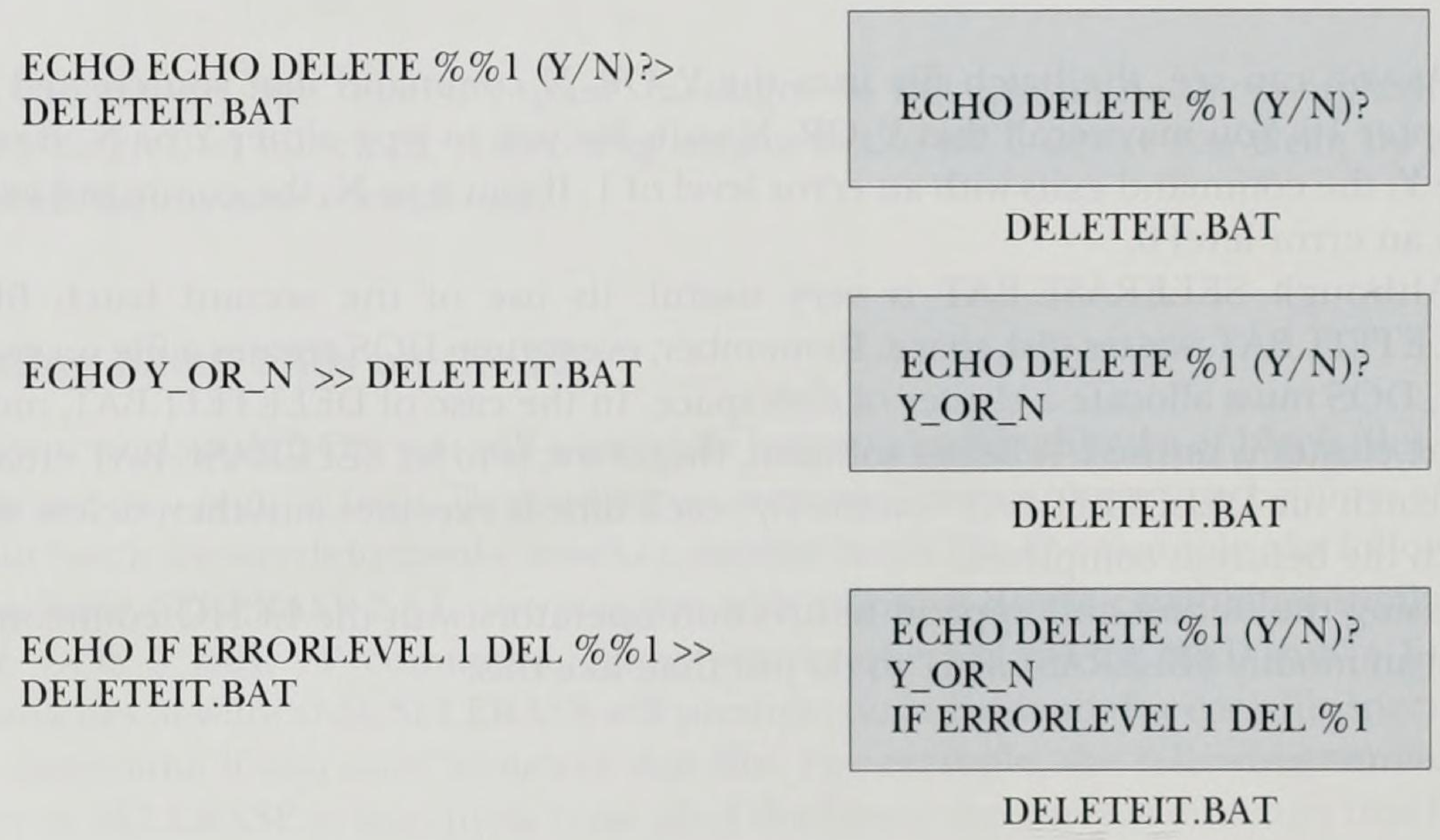
```
ECHO ECHO Delete %%1 (Y/N)? > DELETEIT.BAT
ECHO Y_OR_N >> DELETEIT.BAT
ECHO IF ERRORLEVEL 1 DEL %%1 >> DELETEIT.BAT
REM Loop through the desired file names
:LOOP
  FOR %%I IN (%1) DO CALL DELETEIT %%I
  SHIFT
IF NOT '%1'=='' GOTO LOOP

REM Delete the temporary batch file
DEL DELETEIT.BAT
```

Take a close look at the following commands:

```
REM Build the batch file DELETEIT.BAT
ECHO ECHO Delete %%1 (Y/N)? > DELETEIT.BAT
ECHO Y_OR_N >> DELETEIT.BAT
ECHO IF ERRORLEVEL 1 DEL %%1 >> DELETEIT.BAT
```

The batch file SELERASE.BAT redirects ECHO's output to DELETEIT.BAT, building that batch file. Figure 20-7 illustrates how these commands actually build the batch file.

Note the use of the double percent signs (%%1). As you know, when DOS encounters %1 within a batch file, DOS substitutes for %1 the value of the first batch



```
ECHO ECHO DELETE %%1 (Y/N)?>
DELETEIT.BAT
```

```
ECHO DELETE %1 (Y/N)?
```

```
ECHO Y_OR_N >> DELETEIT.BAT
```

```
ECHO DELETE %1 (Y/N)?
Y_OR_N
```

```
ECHO IF ERRORLEVEL 1 DEL %%1 >>
DELETEIT.BAT
```

```
ECHO DELETE %1 (Y/N)?
Y_OR_N
IF ERRORLEVEL 1 DEL %1
```

Figure 20-7. *Using ECHO to build a batch file "on the fly"*

parameter. In this case, however, we want DOS to ECHO the actual characters "%1". When DOS encounters the double percent signs, DOS only echoes one of them, leaving "%1" as desired.

If you are using DOS 6, the following batch file changes SELERASE.BAT to use the CHOICE command:

```
@ECHO OFF

REM Build the batch file DELETEIT.BAT
ECHO CHOICE Delete %%1 > DELETEIT.BAT
ECHO IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DEL %%1 >> DELETEIT.BAT
REM Loop through the desired file names
:LOOP
  FOR %%I IN (%1) DO CALL DELETEIT %%I
  SHIFT
IF NOT ’%1’==’’ GOTO LOOP

REM Delete the temporary batch file
DEL DELETEIT.BAT
```

If you have existing batch files that call other batch files in this way, you might consider changing the batch files to build their secondary batch files "on the fly."

Key Terms

I/O Redirection The process of routing a program’s source of input from the keyboard to an existing file or directing a program’s output from the screen display to a file or device. DOS provides the input (<), output (>), and append (>>) I/O redirection operations as well as the DOS pipe discussed in Chapter 21.

stdin Abbreviation for the *standard input device*; by default, the keyboard. The input redirection operator lets you change stdin.

stdout Abbreviation for the *standard output device*; by default, the screen display. The output and append redirection operators let you change stdout.

stdprn Abbreviation for the *standard printer device,* which is LPT1—you cannot change stdprn.

stdaux Abbreviation for the *standard auxiliary device,* which is COM1—you cannot change stdaux.

stderr Abbreviation for the *standard error device,* which is the screen display—you cannot change stderr. Many programs write their error messages to stderr to prevent DOS from redirecting them.

Chapter 21

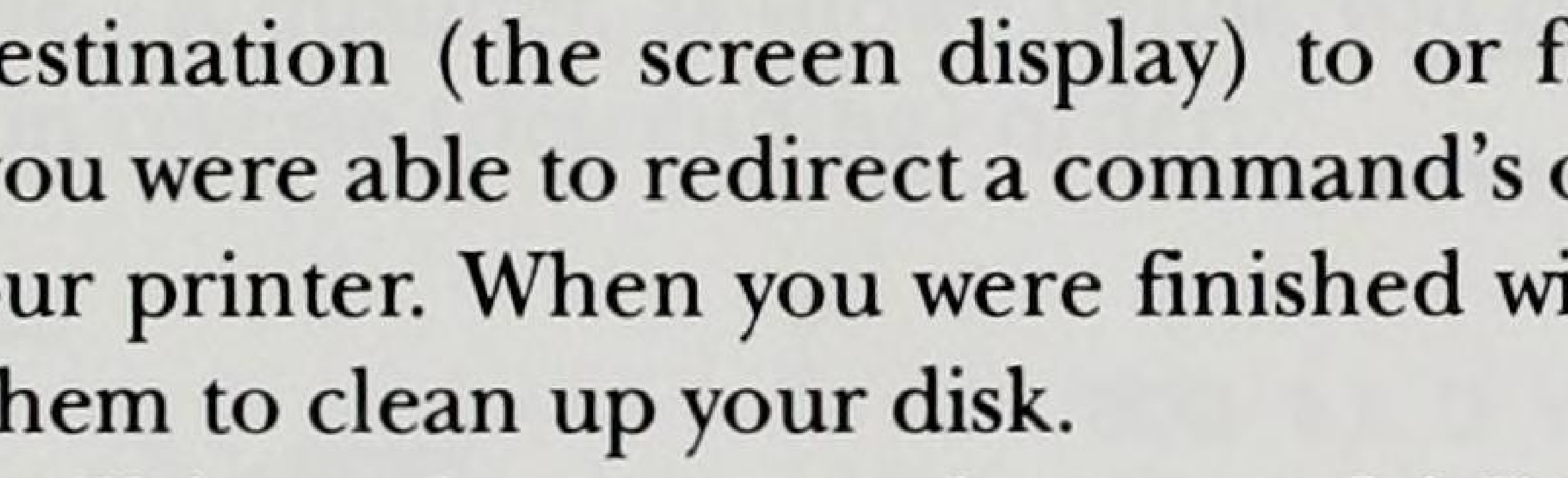
The DOS Pipe

In Chapter 20 you learned how to use the DOS input (<), output (>), and append (>>) redirection operators to change a command's normal source of input (the keyboard) or output destination (the screen display) to or from a file. Using the redirection operators, you were able to redirect a command's output to a temporary file for editing or to your printer. When you were finished with these "temporary" files, you could delete them to clean up your disk.

In this chapter you will learn how to use the powerful DOS pipe operator (|), which combines the benefits and capabilities of both the input and output redirection operators. Using the pipe operator, you can route one command's output into another command's input.

Locating the Pipe Operator on Your Keyboard

Because many books print the DOS pipe operator differently, new users have difficulty identifying the pipe operator on the keyboard. Do not confuse the pipe operator (|) with the exclamation mark (!). The pipe operator typically shares the same key as the backslash key (\) you use in DOS directory names, as shown here:



Hold down either the left or right SHIFT key and then press this key to access the pipe operator. Depending on the book, magazine, or manual you are reading, the pipe character sometimes appears as two small vertical bars (¦) or a solid bar (|). In either case, both symbols refer to the pipe.

Understanding the DOS Pipe

The pipe operator redirects the output of one command into the input of another. For example, in Chapter 20 you used the SORT command to sort a file's contents. Using the pipe operator, the following command redirects the DIR command's output to become the SORT command's input:

```
C:\> DIR  |  SORT <ENTER>
```

Assume the directory contains the following files and subdirectories:

```
C:\SAMPLE> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
 Directory of C:\SAMPLE

.            <DIR>     03-10-93   3:53p
..           <DIR>     03-10-93   3:53p
AAAAAAAA EXT       231 03-10-93   3:54p
BBBBBBBB EXT       128 03-04-93   1:37p
THIRD        <DIR>     03-10-93   3:53p
EEEEEEEE EXT       455 03-10-93   3:55p
DDDDDDDD EXT       128 03-04-93   1:37p
FIRST        <DIR>     03-10-93   3:54p
CCCCCCCC EXT        19 03-10-93   3:54p
SECOND       <DIR>     03-10-93   3:55p
       10 file(s)        961 bytes
                    16403520 bytes free
```

The sorted directory listing then becomes the following:

```
C:\SAMPLE> DIR  | SORT <ENTER>
```

```
                    16399424 bytes free
       10 file(s)        961 bytes
 Directory of C:\SAMPLE
 Volume in drive C is DOS
 Volume Serial Number is 1661-AB75
.            <DIR>     03-10-93   3:53p
..           <DIR>     03-10-93   3:53p
AAAAAAAA EXT      231 03-10-93   3:54p
BBBBBBBB EXT      128 03-04-93   1:37p
CCCCCCCC EXT       19 03-10-93   3:54p
DDDDDDDD EXT      128 03-04-93   1:37p
EEEEEEEE EXT      455 03-10-93   3:55p
FIRST        <DIR>     03-10-93   3:54p
SECOND       <DIR>     03-10-93   3:55p
THIRD        <DIR>     03-10-93   3:53p
```

Pictorially, the pipe redirection operation is shown in Figure 21-1.

The pipe operator is a very powerful tool. By changing the previous SORT command to include the /R qualifier, as shown below, you can quickly display the directory listing sorted in reverse (descending) order:

```
C:\> DIR  |  SORT  /R <ENTER>
```

Likewise, if your directory contains a large number of files, you can use the MORE command (discussed in Chapter 20) to view the files a screenful at a time:

```
C:\> DIR  |  MORE <ENTER>
```

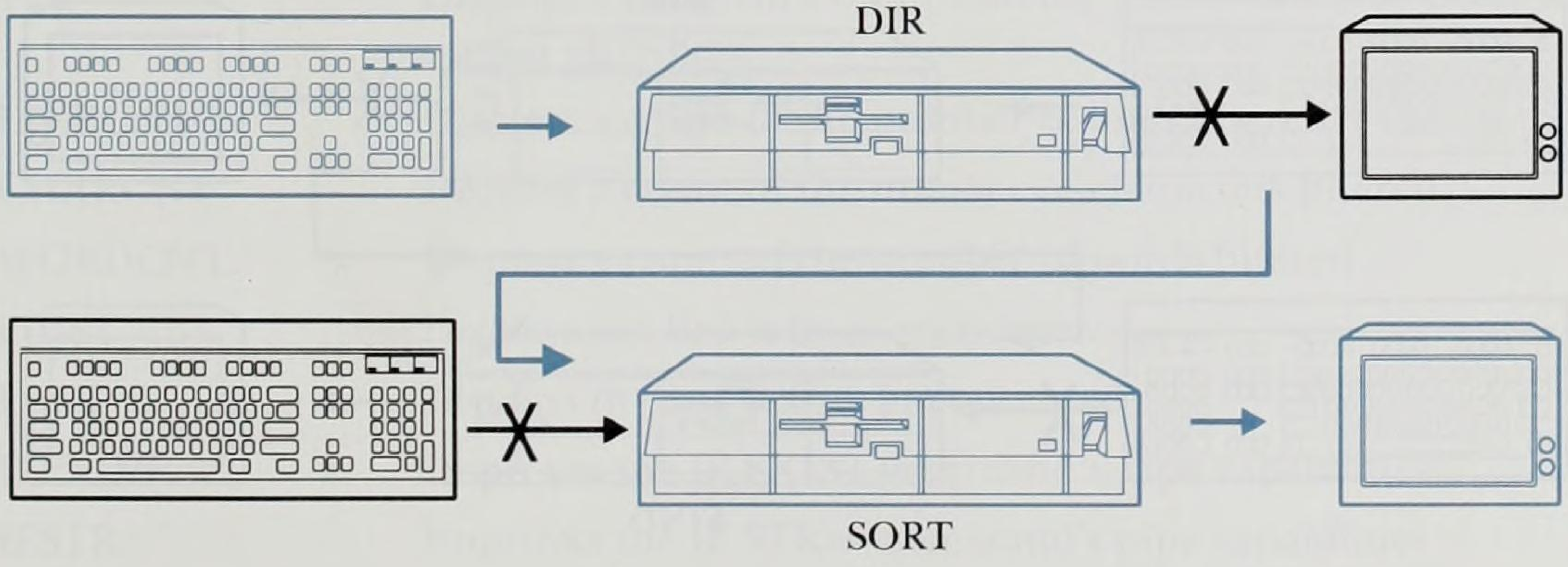


Figure 21-1. Piping the output of DIR into the SORT command

Last, assuming that you only want to display the names of subdirectories that reside in the current directory, you can use the FIND command (Chapter 20) to display only those directory entries containing "<DIR>":

```
C:\SAMPLE> DIR  |  FIND  "<DIR>" <ENTER>
.            <DIR>     03-10-93   3:53p
..           <DIR>     03-10-93   3:53p
THIRD        <DIR>     03-10-93   3:53p
FIRST        <DIR>     03-10-93   3:54p
SECOND       <DIR>     03-10-93   3:55p
```

In this case, the FIND command does not display every line of input it receives. Instead, FIND only displays the lines with "<DIR>" in them. When the pipe operator redirects a command's output through a second command that examines its input and only displays or passes on specific lines, the second command is a *filter*. In general, a filter examines the pipe's contents and lets the data that meet a criteria continue while discarding the data that fail the criteria, as shown in Figure 21-2.

Which Commands Can You Pipe?

As was the case with the output redirection operator, you can use any command that writes its output to the DOS standard output device on the *input side* (left) of a pipe operation. Likewise, just as only a few commands support the input redirection operator (SORT, MORE, and FIND), only the same few commands work on the *filter side* (right) of the pipe operator.

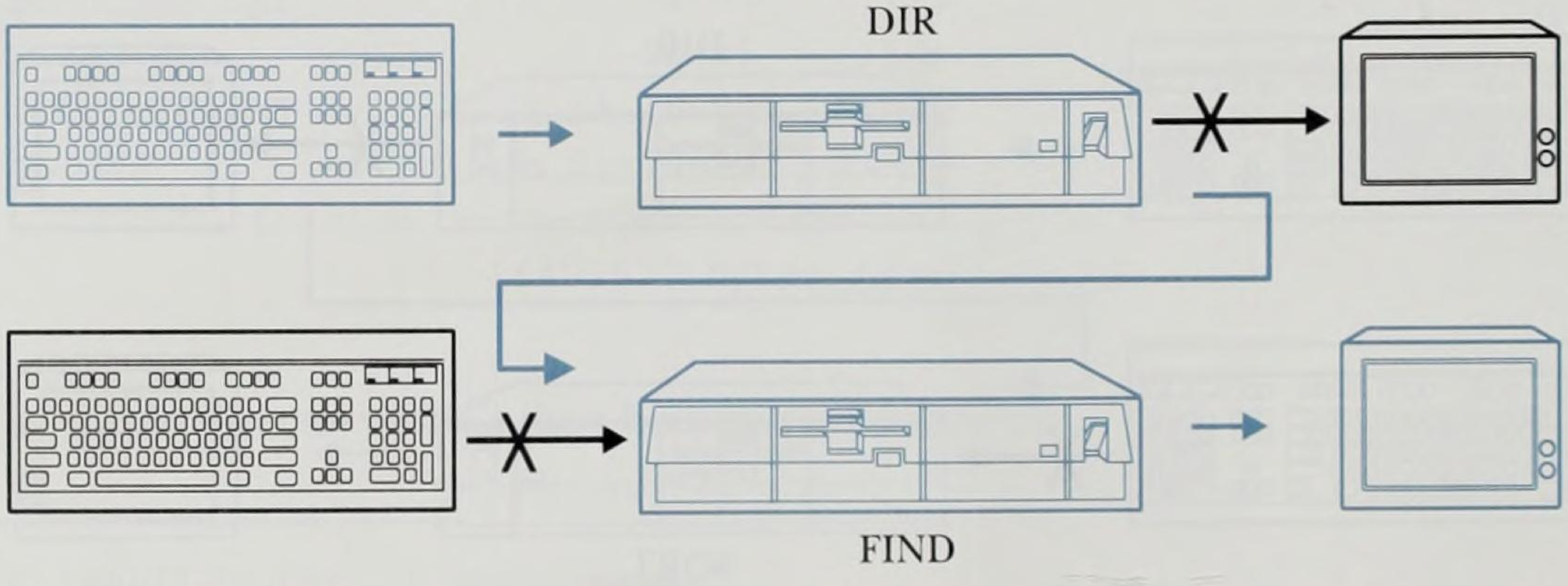


Figure 21-2. Using the pipe operator with a filter

To help you make full use of the pipe, I have assembled a group of pipe utility commands that are available on disk (see the order form at the beginning of this book). Table 21-1 briefly summarizes some of the capabilities these commands provide.

Combining the Pipe and I/O Redirection Operators

In Chapter 20 you experimented with commands that use both the input and output redirection operators, as shown here:

```
C:\> SORT  <  \AUTOEXEC.BAT  >  PRN <ENTER>
```

Just as DOS lets you use both the input and output redirection operators in the same command line, DOS also lets you combine them with the DOS pipe. For example, the following command pipes the output of the DIR command into SORT and then redirects SORT's output to the printer:

```
C:\> DIR  |  SORT  >  PRN <ENTER>
```

Pictorially, the command's redirection is shown in Figure 21-3.

In a similar manner, the following command pipes the output of DIR into SORT, which in turn sorts the directory listing. Rather than displaying the sorted listing to

Command	**Function**
TEE	Displays a program's output on the screen and also pipes the output elsewhere
LINECNT	Displays a count of the number of lines filtered
CHARCNT	Displays a count of the number of characters filtered
WORDCNT	Displays a count of the number of words filtered
FIRST	Displays the first *n* lines encountered
LAST	Displays the last *n* lines encountered
IFEXIST	Improves the IF EXIST command's pipe capabilities
IFSTR	Improves the IF STR== command's pipe capabilities

Table 21-1. Pipe Commands Available through this Book's Disk Offer

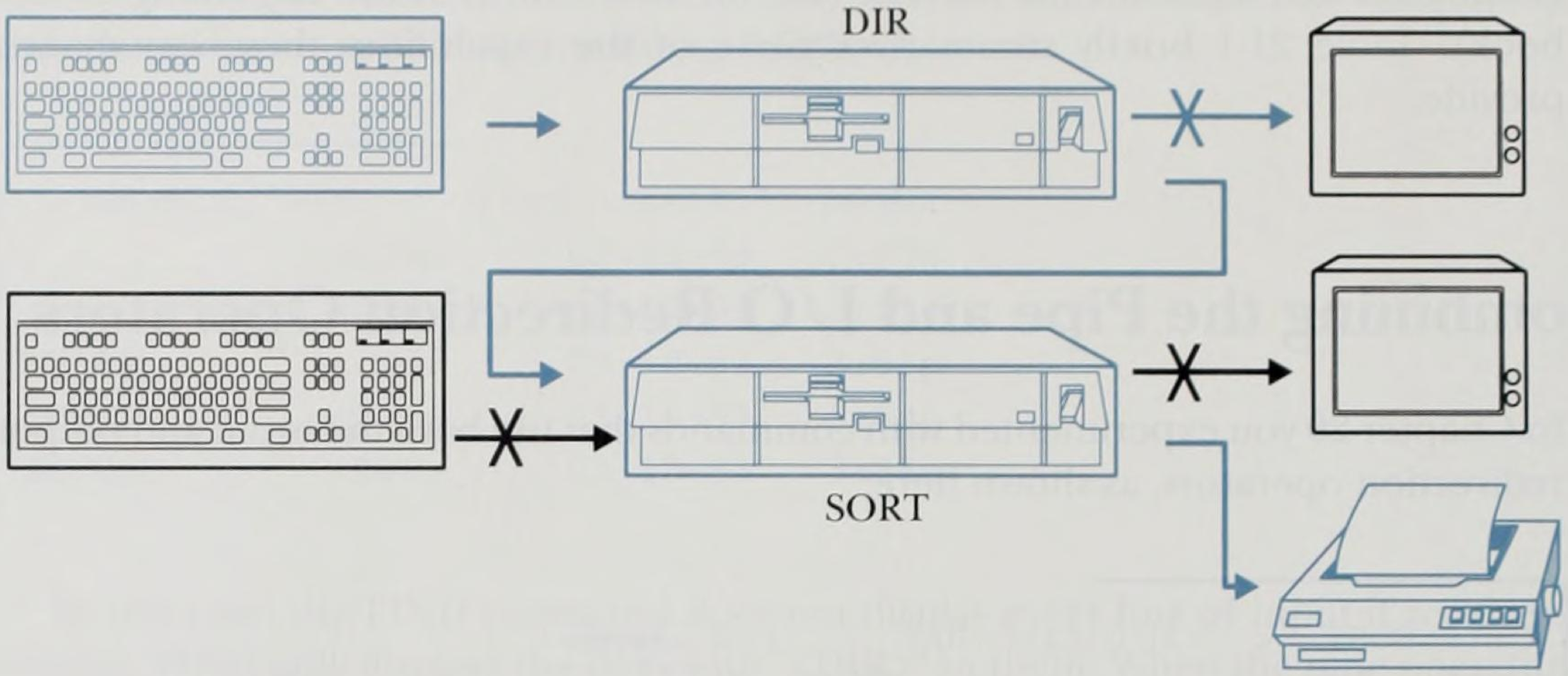


Figure 21-3. *Combining the pipe and output redirection operators*

the screen, DOS pipes SORT's output into MORE, which displays the sorted directory entries a screenful at a time.

```
C:\> DIR | SORT | MORE <ENTER>
```

As before, pictorially, the redirection is shown in Figure 21-4.

In general, the only restriction to the number of pipe operators you can place in a single command is the 127-character limit on command length.

Pipe Operator Pitfalls

A difficult pipe error to detect occurs when you pipe the output of a command that prompts for user input and then waits for you to type in the input before it continues. When this occurs, your system will appear to hang up. Nothing will appear on the screen except for a blinking cursor. For example, as discussed in Chapter 4, the DIR /P command directs DIR to pause with each screenful of filenames, prompting you to press any key to continue. If you redirect DIR /P into the SORT command for example, your system may appear to hang because DIR is waiting for a keyboard response. Because DOS also redirects the prompt to press a key to continue into SORT, you may think the system is hung up. If this occurs, use CTRL-C to cancel the command. If your system remains frozen up, press CTRL-ALT-DEL to restart the system.

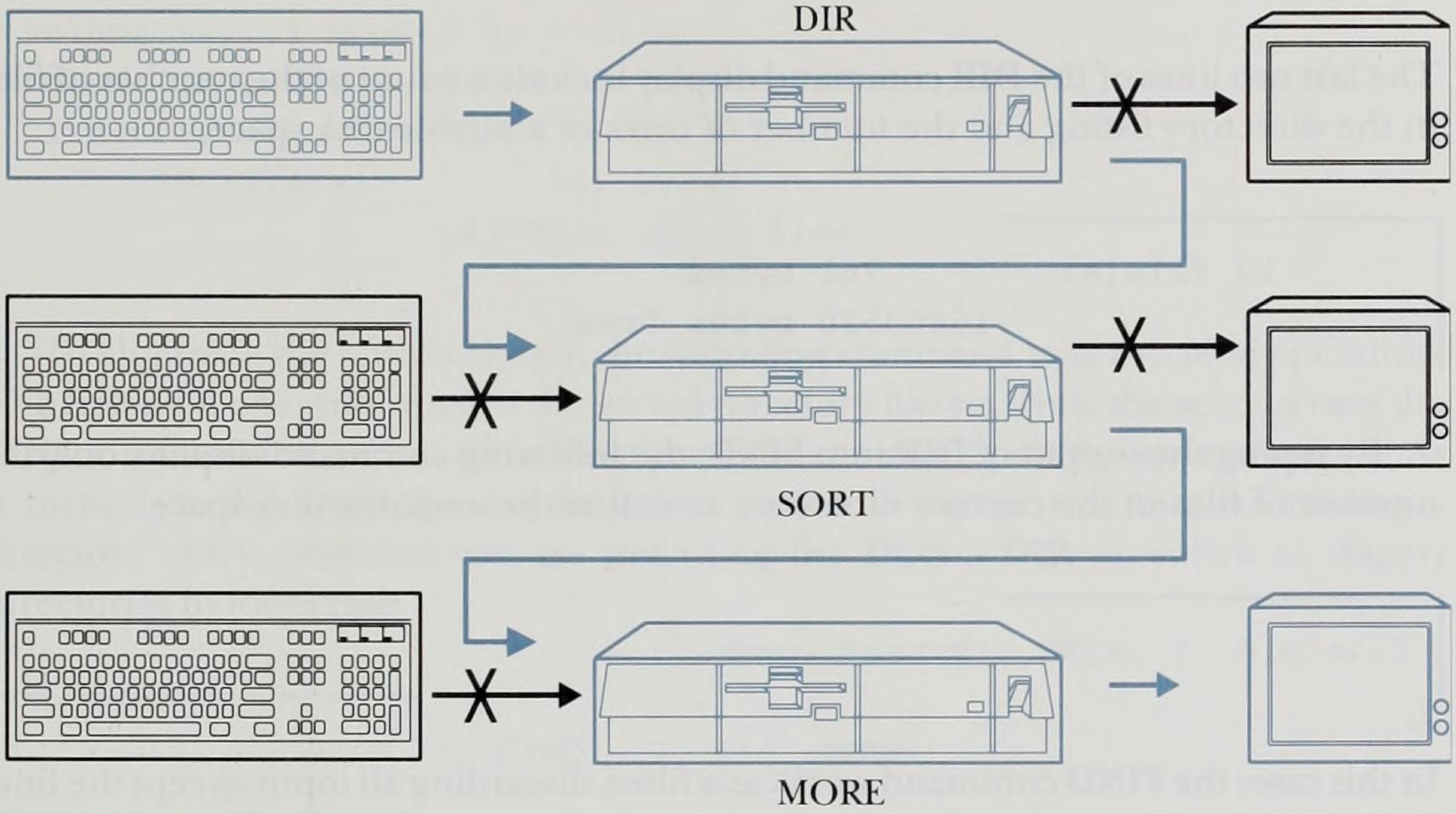


Figure 21-4. *Combining two pipe operations in the same command*

SUMMARY

DOS Pipe Operator (|)

The pipe operator combines the capabilities of the input and output redirection operators, letting you direct one command's output to become a second command's input. The following command pipes the output of the TREE command into the MORE command to display the disk's current directory structure (and files) a screenful at a time:

```
C:\> TREE C:\ /F | MORE <ENTER>
```

When the second command in the pipe operation selectively displays or discards lines of the piped input, the second command is called a filter.

Hands On

The last two lines of the DIR command display include a count of the number of files in the directory listing and the number of bytes of available disk space.

```
       10 file(s)          961 bytes
                      16403520 bytes free
```

By piping the output of DIR into FIND, the following command displays only the number of files in the current directory, as well as the available disk space:

```
C:\> DIR | FIND "bytes" <ENTER>
```

In this case, the FIND command works as a filter, discarding all input except the lines containing the word "bytes."

Actually, because DOS filenames prior to DOS 5 (see Chapter 4 for DIR's /L lowercase switch) only appear in uppercase letters, you can reduce the previous command to only search for the lowercase letter b, as shown here.

```
C:\> DIR | FIND "b" <ENTER>
```

As presented earlier, the following command uses the pipe operator to list the files in the current directory sorted in reverse order.

```
C:\> DIR | SORT /R <ENTER>
```

Unfortunately, because SORT cannot distinguish the filenames from the additional directory information, the command's output may become messy:

```
C:\SAMPLE> DIR | SORT /R <ENTER>
THIRD        <DIR>      03-10-93   3:53p
SECOND       <DIR>      03-10-93   3:55p
FIRST        <DIR>      03-10-93   3:54p
EEEEEEEE EXT        455 03-10-93   3:55p
DDDDDDDD EXT        128 03-04-93   1:37p
CCCCCCCC EXT         19 03-10-93   3:54p
BBBBBBBB EXT        128 03-04-93   1:37p
AAAAAAAA EXT        231 03-10-93   3:54p
             <DIR>      03-10-93   3:53p
```

```
..          <DIR>       03-10-93    3:53p
 Volume Serial Number is 1661-AB75
 Volume in drive C is DOS
 Directory of C:\SAMPLE
       10 file(s)          961 bytes
                      16399424 bytes free
```

To eliminate the screen clutter, the following command uses two pipe operators. The first pipe operator obtains the sorted directory listing, while the second uses the FIND command as a filter for all the lines not containing the lowercase letter e. As it turns out, the lowercase e is found in each line except the lines containing a directory entry, provided you are not using the DOS 5 DIR /L switch to display directories in lowercase.

```
C:\> DIR  |  SORT  |  FIND  /V  "e" <ENTER>
```

Given the previous directory contents, the command's output would become the following:

```
C:\> DIR  |  SORT  |  FIND  /V  "e" <ENTER>
.             <DIR>       03-10-93   3:53p
..            <DIR>       03-10-93   3:53p
AAAAAAAA EXT         231 03-10-93   3:54p
BBBBBBBB EXT         128 03-04-93   1:37p
CCCCCCCC EXT          19 03-10-93   3:54p
DDDDDDDD EXT         128 03-04-93   1:37p
EEEEEEEE EXT         455 03-10-93   3:55p
FIRST         <DIR>       03-10-93   3:54p
SECOND        <DIR>       03-10-93   3:55p
THIRD         <DIR>       03-10-93   3:53p
```

As you can see, FIND restricted the output to only those lines not containing the lowercase e.

As you have just found, by piping the DIR command into SORT, you can display a directory sorted by name. There may be times, however, when you want to display the directory listing sorted by extension or file size. The SORT command, discussed in Chapter 20, lets you specify a column on which to sort the input data. The following command, for example, displays the directory entries sorted by extension (which begins in column 10):

```
C:\> DIR  |  SORT  /+10  |  FIND  /V  "e" <ENTER>
```

Likewise, the following command displays the entries sorted by size (column 13):

```
C:\> DIR | SORT /+13 | FIND /V "e" <ENTER>
```

By changing the previous command to include the output redirection operator, you can print the names of the files in the current directory sorted by size:

```
C:\> DIR | SORT /+13 | FIND /V "e" > PRN <ENTER>
```

Review

1. What is the function of the pipe operator?
2. What restricts the number of pipe operators that can be placed in one command line?
3. Use the pipe operator and the FIND command to display only the names of files that reside in the current directory, discarding subdirectory names and other information (DOS 5 or later users, be aware of DIR /L).
4. What is a filter?
5. Pictorially represent the following command redirection:

   ```
   C:\> SORT < \AUTOEXEC.BAT | MORE <ENTER>
   ```

6. What is the function of the following command?

   ```
   C:\> TREE C:\ /F | FIND ".BAT" | MORE <ENTER>
   ```

Answers

1. The pipe operator combines the capabilities of the input and output redirection operators. In other words, the pipe redirects one command's output to become the input of another.

2. In general, the only restriction on the number of commands you can pipe together is the 127-character limit on the length of the command line. If you are using the pipe with the IF command, however, you can only use the pipe operator once.
3. The pipe operator and FIND command are used together to create the directory display like this:

```
C:\> DIR | FIND /V "<DIR>" | FIND /V "e" <ENTER>
```

4. A filter is a command appearing to the right of a pipe operator that examines its input and selectively discards or passes on to the next program (or to the screen display) only those lines that meet the desired criteria.
5. The pictorial representation of the following command redirection is shown in Figure 21-5:

```
C:\> SORT < \AUTOEXEC.BAT | MORE <ENTER>
```

6. The command displays the names and directory of each batch file on your disk a screenful at a time.

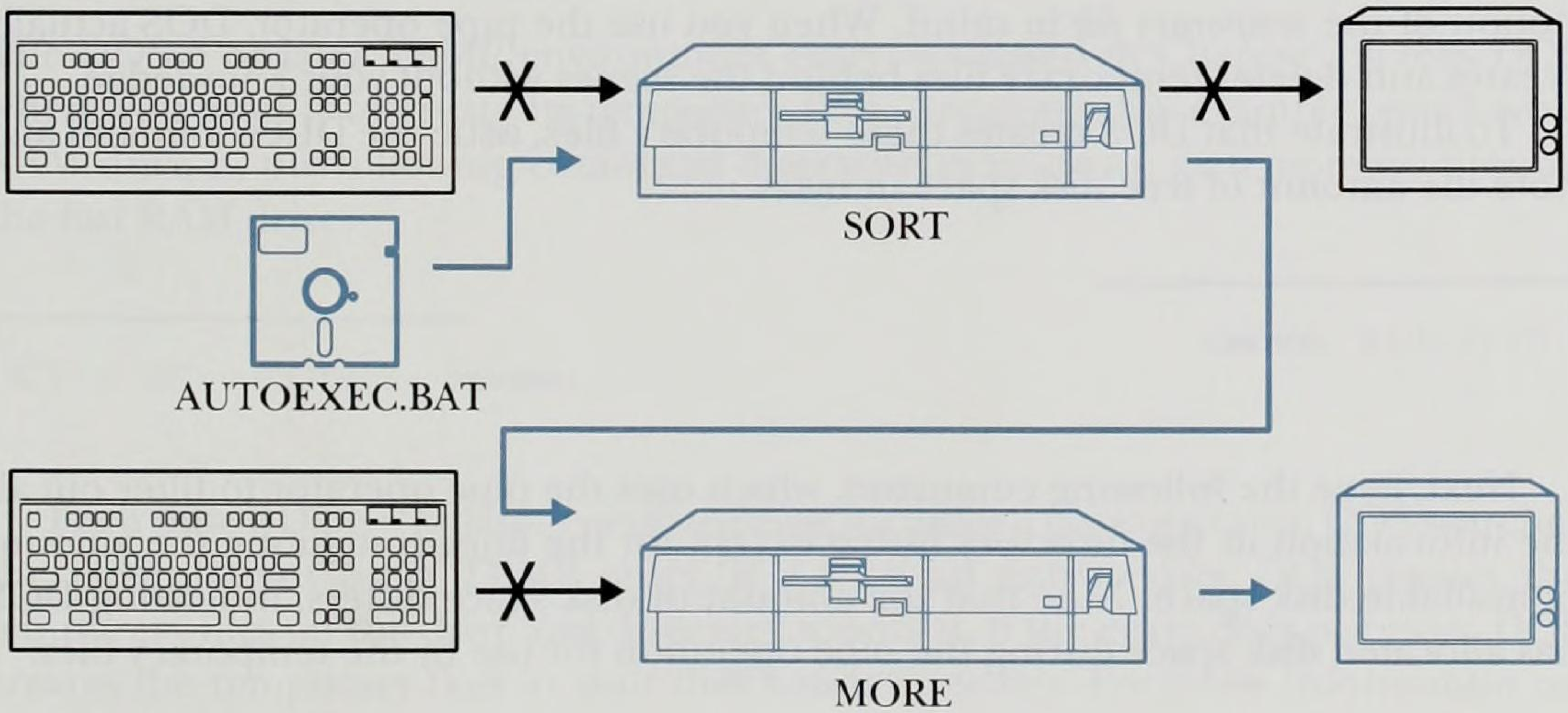


Figure 21-5. *Sorted output of AUTOEXEC.BAT is piped to MORE, which displays it a screenful at a time*

Advanced Concepts

As you have learned, the pipe operator lets you direct the output of one command to become the input of another. In general, the pipe operator combines the input and output redirection operations. For example, the following command uses the pipe to display a sorted directory listing:

```
C:\> DIR  |  SORT <ENTER>
```

In order to achieve the same result using the input and output operators, you can create a temporary file named TEMP and redirect its output to SORT, as shown here:

```
C:\> DIR  >  TEMP <ENTER>
C:\> SORT  <  TEMP <ENTER>
```

When the SORT command completes, you can delete the temporary file.

```
C:\> DEL  TEMP <ENTER>
```

You can see that the piped operation performed in one command the steps that required three commands with the redirection operators.

To understand how DOS implements the pipe operator, you need to keep the notion of the *temporary file* in mind. When you use the pipe operator, DOS actually creates and deletes temporary files behind the scenes without your knowledge.

To illustrate that DOS creates these temporary files, issue the DIR command and note the amount of free disk space in bytes.

```
C:\> DIR <ENTER>
```

Next, issue the following command, which uses the pipe operator to filter out all the information in the directory listing except for the lines containing the amount of available disk space. Note that the amount of disk space differs, indicating DOS has allocated disk space during the pipe operation for use by the temporary files.

```
C:\> DIR | FIND "bytes" <ENTER>
```

When the command completes, repeat the DIR command and note that the free disk space again matches the original, indicating DOS has automatically deleted the temporary file at the completion of the pipe operation.

Depending on your DOS version, DOS may create the temporary file in the current directory or the root directory.

You should understand that the pipe operator creates temporary files for several reasons. First, if the current disk is write protected, DOS cannot create the temporary file, and the command will fail.

```
Write protect error writing drive A
Abort, Retry, Fail?
```

Likewise, if the current disk does not have enough disk space for the temporary files, or the root directory is full, commands using the pipe operator will fail.

```
Intermediate file error during pipe
```

Last, if you are using the pipe operator in a command that requires maximum processing performance, you might consider selecting a RAM drive as the default. Because of the RAM drive's fast speed, DOS can create the temporary file much faster, which helps the command complete sooner.

The DOS 5 or later TEMP Environment Entry

The DOS 5 or later TEMP environment entry, discussed in Chapter 19, tells DOS where you want it to create its temporary files. Assuming, for example, you have a RAM drive E, the following command directs DOS to create its temporary files on the fast RAM drive:

```
C:\> SET  TEMP=E: <ENTER>
```

Each time a DOS 5 or later command line includes a pipe operator, DOS searches the environment for a TEMP entry. If a such an entry exists, DOS creates the temporary files on the drive and directory specified. If the entry does not exist, DOS creates the temporary files in your disk's root directory. For more information on RAM drives, see Chapter 32.

Batch File Pipe Restrictions

In general, DOS only restricts the number of pipe operations you can perform in one command to the maximum command length of 127 characters. The only exception to the essentially unlimited use of pipe operators is when you combine the pipe with the IF command. IF, discussed in Chapter 16, lets your batch files perform conditional processing. The following batch file SORTIT.BAT examines the batch parameter %1. If %1 is A, the batch file displays the directory listing in ascending order. Likewise, if %1 is D, the batch file uses descending order.

```
@ECHO OFF
IF '%1'=='A' DIR | SORT
IF '%1'=='D' DIR | SORT /R
```

As you found in the directory sort examples presented in the beginning of this chapter, if you don't filter out the lines that don't contain filenames using FIND, the directory listing displayed can become quite messy. As an initial solution to this problem, you might simply add the FIND command to the batch file, for example:

```
@ECHO OFF
IF '%1'=='A' DIR | SORT | FIND /V "e"
IF '%1'=='D' DIR | SORT /R | FIND /V "e"
```

Unfortunately, each time you execute the batch file, DOS will now display the "File creation error" message, and the batch file will fail (some DOS versions won't display the error message). The error occurs because when you use the IF command, DOS restricts you to only using the pipe operator once. To correct this situation, you must restructure your batch file to put the alternative uses into separate subroutines, as shown here. BAD_PARAM is for any keypress except A or D.

```
@ECHO OFF
IF '%1'=='A' GOTO ASCENDING
IF '%1'=='D' GOTO DESCENDING
GOTO BAD_PARAM

:ASCENDING
DIR | SORT | FIND /V "e"
GOTO DONE

:DESCENDING
DIR | SORT /R | FIND /V "e"
GOTO DONE

:BAD_PARAM
ECHO Use A for Ascending or D for Descending
:DONE
```

Key Terms

Pipe operator (|) A command-line redirection operator that directs the output of one command to become the input of a second command. The pipe operator combines the capabilities of the input and output redirection operators.

Filter A command that appears in the command line to the right of the pipe operator whose purpose is to only output lines meeting a specific criteria, while discarding all others.

Chapter 22

Protecting Your Files with Backups

No matter how well you treat your computer, some of its parts will eventually wear out. If your hard disk fails, you may lose all the information the disk contains. Your only way of recovering your disk's previous contents is from backup copies stored on floppy disks. If you are using DOS 5 or later, the DOS crisis-prevention utilities UNDELETE and UNFORMAT, discussed in Chapter 28, may help you recover from a DOS command that accidentally destroys information on your disk. If you are not using DOS 5 or later, many third-party software packages provide similar capabilities. Unfortunately, even the best file recovery utilities can't guarantee 100 percent recovery of your disk—you may still lose key information.

If you are using DOS 5 or earlier, this chapter examines the BACKUP command, which copies your hard disk files to floppy disks for safe storage. Should your hard disk fail or an accidental DOS command such as FORMAT destroy the information on your disk, you can restore the backup copies of the damaged files from floppy disks.

If you are using DOS 6, this chapter presents the MSBACKUP utility, a very powerful menu-driven program. If you have not been performing backup operations in the past, MSBACKUP's speed and flexibility may motivate you to do so.

To fully protect your files, you need to perform backups on a regular basis. Using the concepts presented in this chapter, you will find that your backup operations should only take a few minutes each day.

Understanding the Backup Process

Whether you use the BACKUP command or a third-party backup utility, the backup process is essentially the same. You will begin with a set of unused floppy disks. Depending on the amount of hard disk space in use and the size of your floppy disks, the number of floppy disks you'll need will differ. Chapter 25 discusses the CHKDSK command in detail. For now, invoke CHKDSK from the DOS prompt to determine the amount of space your hard disk is using.

```
C:\> CHKDSK <ENTER>

Volume DOS 6 DISK    created Apr 24, 1993 8:23p
Volume Serial Number is 166F-AD7B

   21309440 bytes total disk space
      53248 bytes in 3 hidden files
      47104 bytes in 12 directories
    4978688 bytes in 362 user files
      55296 bytes in bad sectors
   16175104 bytes available on disk

       2048 bytes in each allocation unit
      10405 total allocation units on disk
       7898 available allocation units on disk

     655360 bytes total memory
     527488 bytes free
```

Depending on your DOS version, the information CHKDSK displays may differ slightly.

To determine the amount of space in use, subtract the amount of available disk space from the amount of total disk space.

21,309,440 bytes total disk
–16,175,104 bytes available on disk

5,134,336 bytes in use

Next, divide the amount of hard disk space in use by the size of your floppy disk. Table 22-1 describes the common floppy disk sizes.

Disk Type	Data Storage in Bytes
360Kb	362,496
720Kb	730,112
1.2Mb	1,213,952
1.44Mb	1,457,664

Table 22-1. *Common Floppy Disk Storage Capacities*

Using CHKDSK's previous output and 1.2Mb floppy disks, the number of floppy disks you would need to back up this disk becomes the following:

$$\begin{aligned}\text{Number of 1.2Mb disks} &= 5{,}134{,}336 \div 1{,}213{,}952\\ &= 4.23\\ &= 5\end{aligned}$$

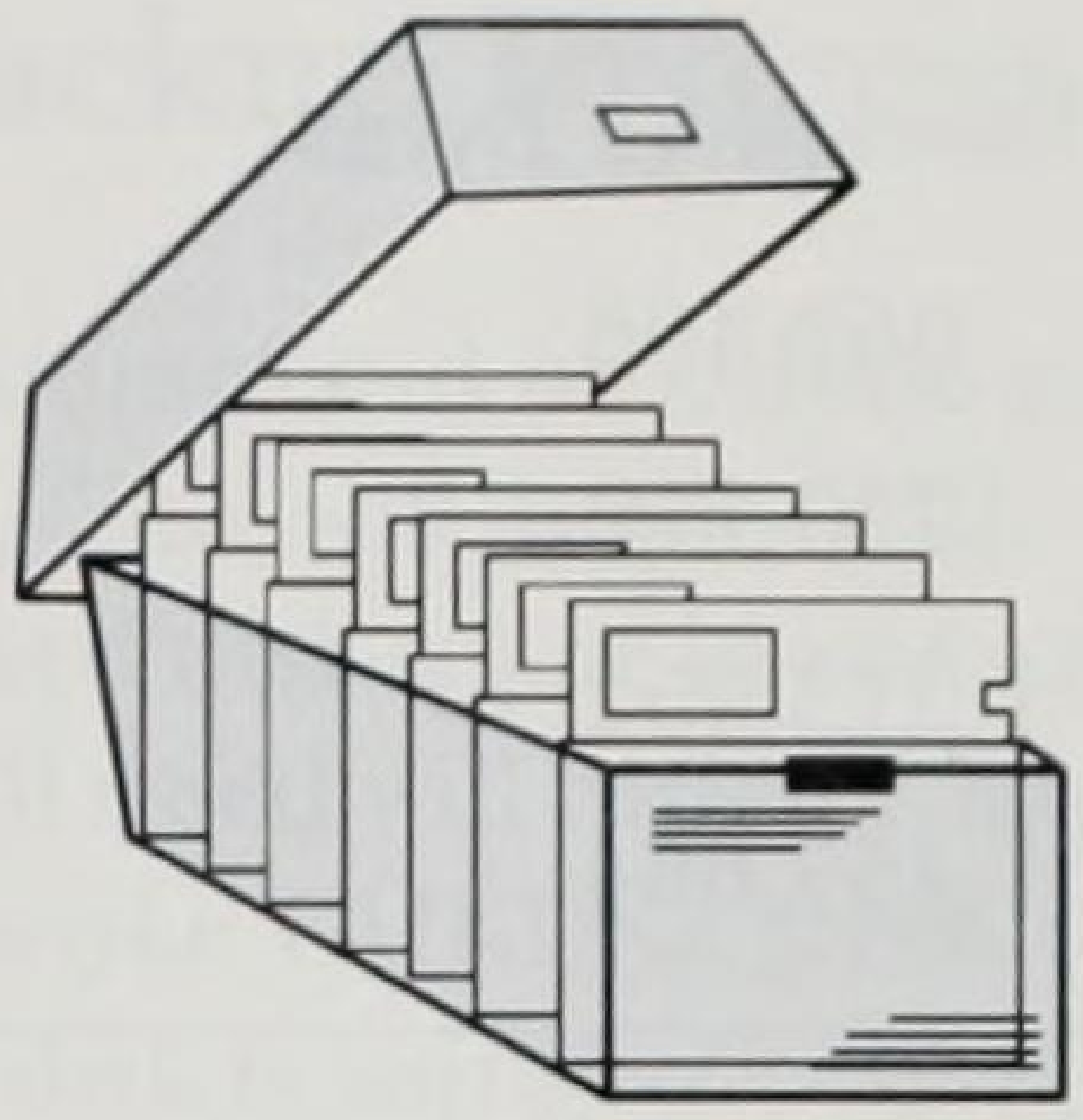
Different DOS versions let you direct BACKUP to format the floppy disks as part of the process. Unfortunately, using BACKUP to format the disks increases the amount of time your backup operation requires. Most users avoid performing backups if the operation takes too long. If you have enough formatted floppy disks available, backup operations proceed quite quickly.

Do not format your floppy disks using the /S switch. As you read in Chapter 7, the /S switch directs FORMAT to copy the DOS startup files to the floppy disk. Since you won't be using your backup disks to boot DOS, placing these system files on the disk would consume disk space unnecessarily.

To back up your hard disk files, you will perform two types of backup operations. First, once a month you will back up your entire hard disk. This backup operation will copy every file on your hard disk to floppy disks. Depending on the size and number of files on your hard disk, this backup operation may take an hour or more. Because the complete hard disk backup operation takes so much time and uses so many floppy disks, you will only perform the complete disk backup once a month.

You should perform the second backup operation daily. This backup operation copies to floppy disks only those files you have created or changed that day. Because most users only create or change a few files each day, this second backup operation normally only takes one or two disks and a few minutes of your time. Many users won't perform these backup operations each day. However, by not backing up your files on a regular basis, you leave open a window for possible loss of files or key information.

When you complete your backup operation, you will have a collection of floppy disks that contain your file backups. For safety, place these disks into a floppy disk storage container, as illustrated on the following page. Such containers are inexpensive to buy and will protect the disks from dust and bending or folding. It is a good idea to have a separate box for backups only.

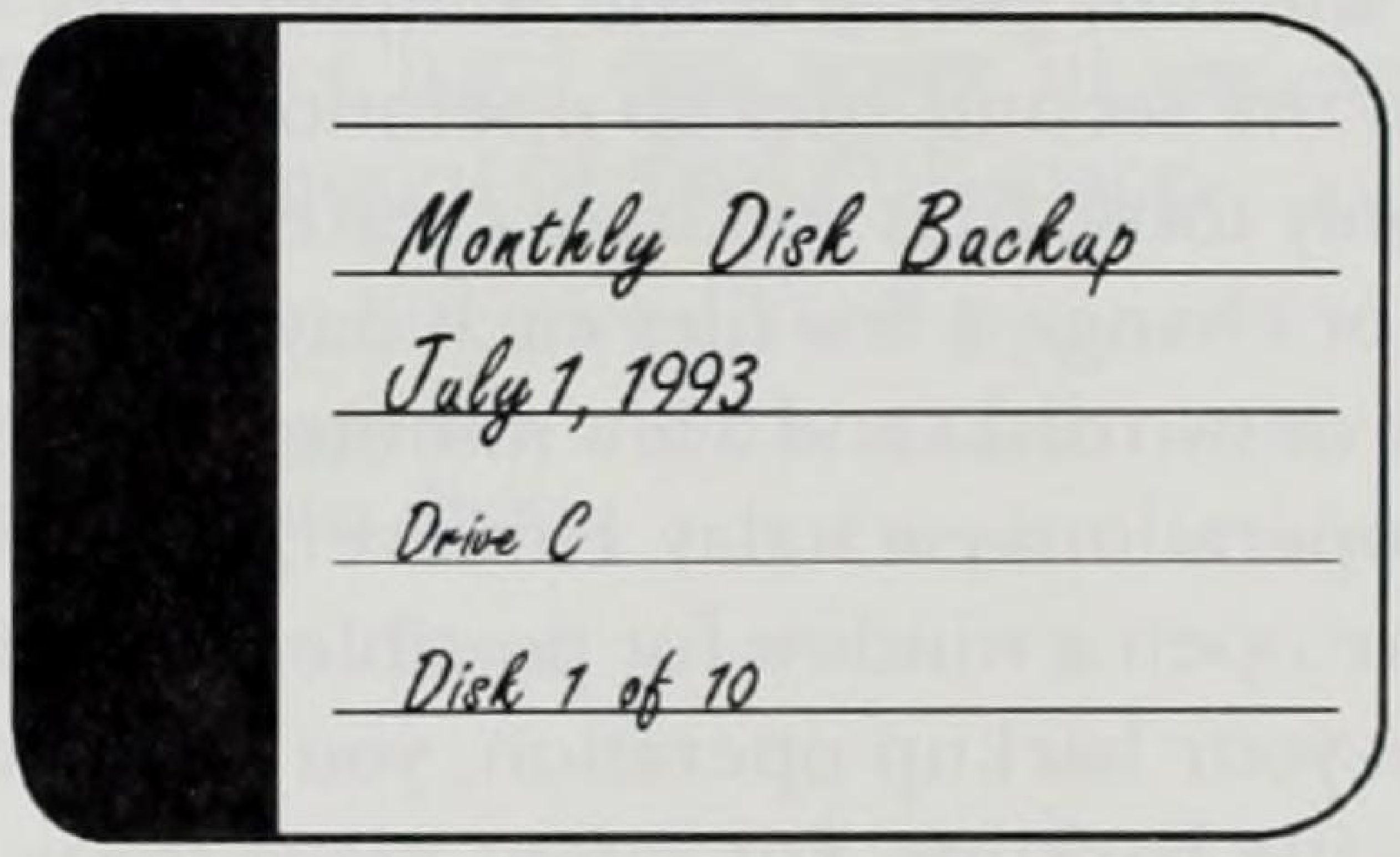


The BACKUP command is an external command that resides in your DOS directory. BACKUP's counterpart is RESTORE, also an external DOS command. Should you ever need to copy (restore) one or more files from your backup floppy disks, you will use the RESTORE command, discussed later in this chapter.

Backing Up Your Entire Hard Disk

You should get into the habit of backing up all the files on your hard disk once a month. When you successfully back up your entire hard disk, you can remove the current floppy disk backups from your disk storage container, replacing them with your new backup disks. You can now use the previous floppy disks for other operations. Before you begin, use the CHKDSK command as previously discussed to determine the number of floppy disks the backup operation will require. In DOS 4 and later, if the floppy disks are not formatted, you can use the /F switch to direct BACKUP to format them. Using BACKUP to format the floppy disks increases the amount of time your backup operations require. For more information on /F, refer to the Command Reference at the end of this book.

A complete hard disk backup may require many floppy disks. Each time BACKUP fills a floppy, make sure you place on the floppy a disk label that specifies that the disk contains files from a hard disk backup. The label should include the backup date, and the disk number, as shown in this example:



Monthly Disk Backup
July 1, 1993
Drive C
Disk 1 of 10

Use the following BACKUP command to back up your entire hard disk:

```
C:\> BACKUP  C:\*.*  A:  /S <ENTER>
```

The command uses the asterisk wildcard to direct BACKUP to copy every file beginning in the root directory. The /S switch directs BACKUP to copy files in each directory below the directory specified. Because the command specifies the root directory and /S, BACKUP will copy your entire directory tree. Drive A, in this case, is the target drive to which it will copy the files.

The BACKUP command begins by prompting you to place the first backup floppy disk in drive A, then warns you that it will erase any files in the disk's root directory.

```
Insert backup diskette 01 in drive A:

WARNING! Files in the target drive
A:\ root directory will be erased
Press any key to continue . . .
```

Place your first backup floppy disk in drive A and press ENTER. BACKUP will respond with a message stating that it is backing up files to disk 1 in drive A. BACKUP will also display the name of each file it successfully copies to the floppy disk.

```
*** Backing up files to drive A: ***
Diskette Number: 01

\AUTOEXEC.BAT
\CONFIG.SYS
\COMMAND.COM
\DOS\ANSI.SYS
\DOS\CHKDSK.COM
```

Eventually the floppy disk in drive A will fill. BACKUP will display the following prompt for you to insert the second floppy disk into drive A.

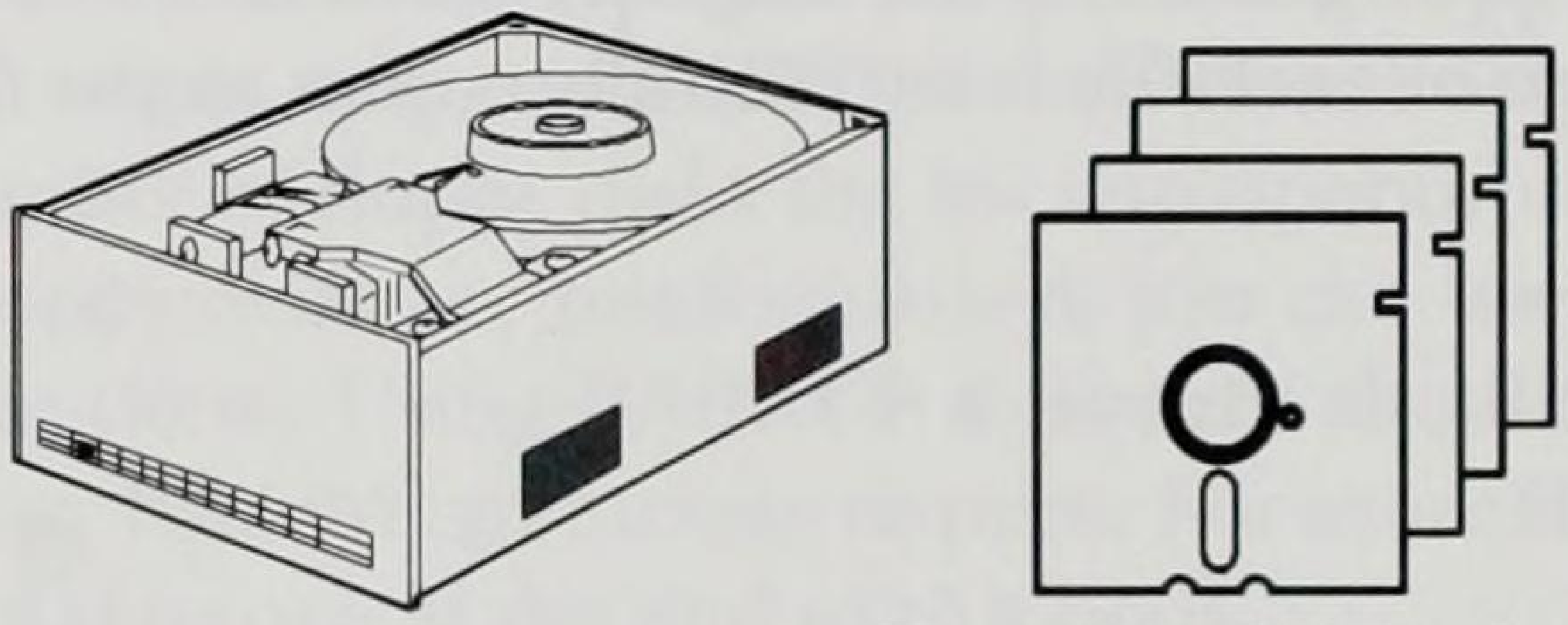
```
Insert backup diskette 02 in drive A:

WARNING! Files in the target drive

A:\ root directory will be erased
Press any key to continue . . .
```

Remove the first floppy disk from drive A and place the disk in its disk envelope. Next, attach a label to the disk describing the disk's contents. Remember, never write on a label already attached to a disk. In doing so, you might damage the disk, destroying the disk's contents. Next, place the second floppy disk in drive A and press ENTER to resume the backup operation.

As each floppy disk fills, repeat the process of removing, labeling, and inserting new disks. When the backup operation completes, place these floppy disks in a safe storage location.

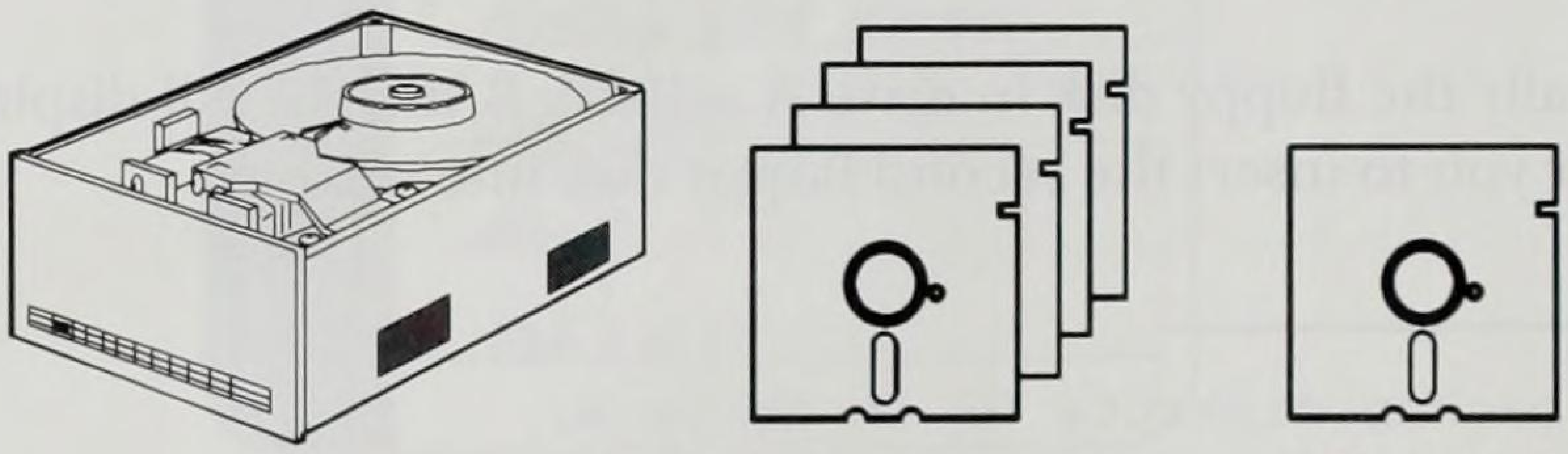
Performing Daily Backup Operations

Because backing up every file on your hard disk can consume so much time, BACKUP also lets you only back up those files you have created or changed since the last backup operation. These partial backup operations are called *incremental backups.* Because most users typically only change or create a few files each day, incremental backup operations are very fast. Here's how incremental backups keep your backup files up to date. Assume, for example, that on January 1 you perform a complete backup of your disk as previously discussed. BACKUP will produce a current backup of every file on your disk, as illustrated here:



Complete disk backup

On January 2, you create the file NEW.DAT and change the contents of the files ONE.DAT and TWO.DAT. Because it only copies these three files, your incremental backup completes very quickly. Your backup disks now contain the files as shown in the following:

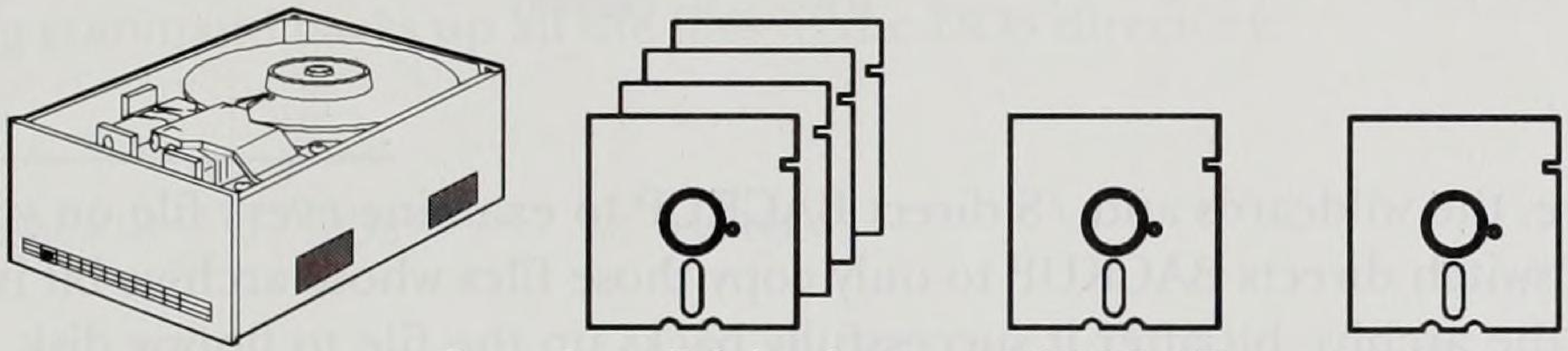


Complete disk backup

NEW.DAT
ONE.DAT
TWO.DAT

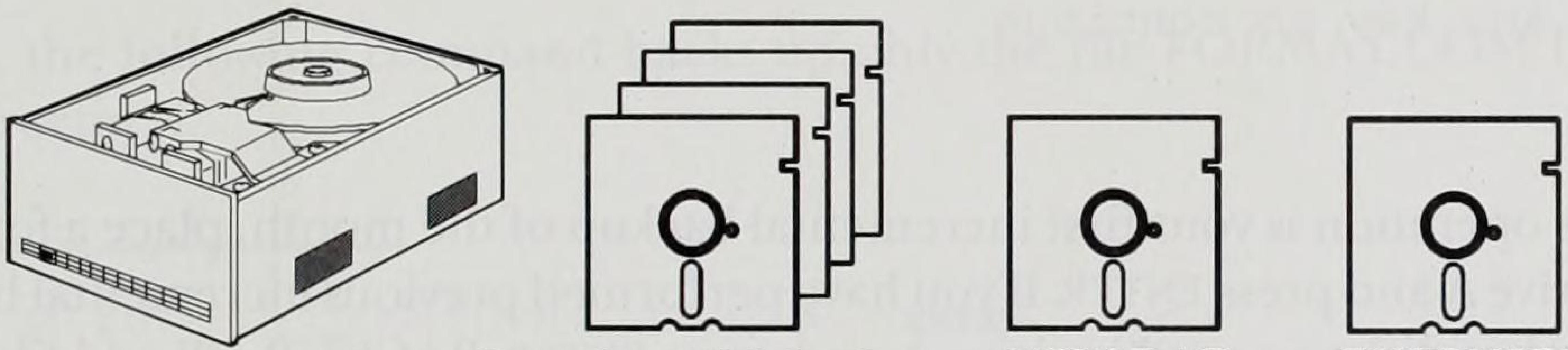
The complete disk backup copies every file on your disk, while the incremental backup copies new files and changes.

On January 3, you create two large files, BUDGET.DAT and EXPENSES.DAT. The first file fits on your first incremental backup disk, while the second is placed on a second backup disk.



You can see that, during the month, your number of incremental backup disks will grow.

On January 4, you change the contents of the file NEW.DAT. Your incremental backup will copy the file's contents, as shown here:



Throughout the month, different versions of a file might be on several disks, depending on how often you change the file's contents and perform your backups.

By January 31, you will have the floppy disks from the complete disk backup, plus disks containing the incremental backups. As long as you perform backup operations at the end of each day, the most work you can lose due to a disk failure is the current day's files.

To perform incremental backup operations, the BACKUP command must be able to determine which files are new and which files you have changed. As you have learned, a directory is a list of filenames, extensions, sizes, and date and time stamps. In addition, DOS keeps a byte of information for each entry, called the file's *attribute byte.* Each time you create a file or change a file's contents, DOS sets a value in the attribute byte called the *archive bit.* The archive bit tells DOS that the file's current contents have not been archived or backed up to floppy disk with BACKUP. When you use BACKUP to perform an incremental backup operation, it examines every file's archive bit. If the bit is *set,* BACKUP knows the file needs archiving. As a result, BACKUP copies the file to the floppy disk. After the file is successfully copied to a floppy, BACKUP *clears* the archive bit.

The number of floppy disks your incremental backup requires depends on the number of files you have changed or created and their sizes. Use the following BACKUP command to perform an incremental backup:

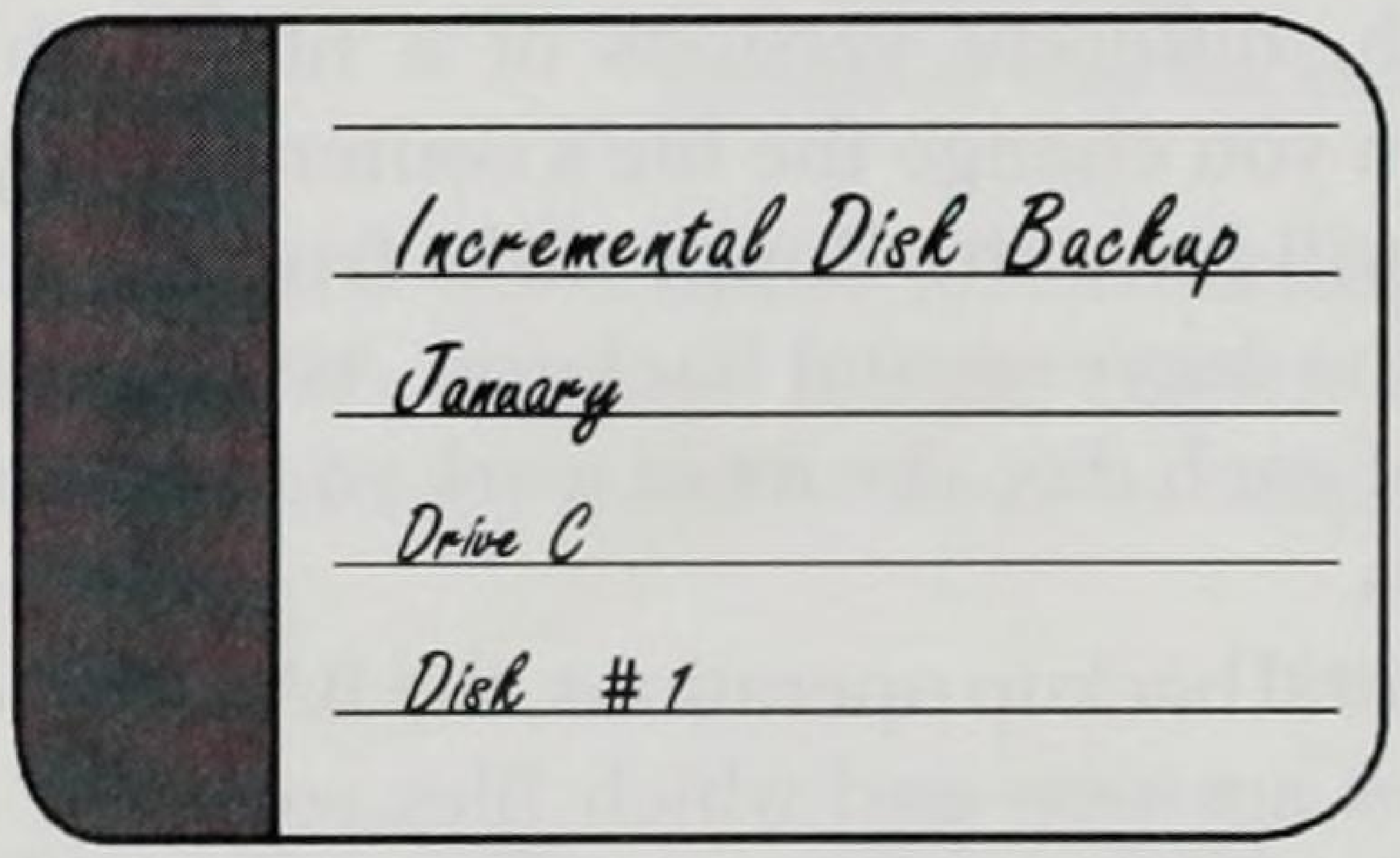
```
C:\> BACKUP  C:\*.*  A:  /S  /A  /M <ENTER>
```

As before, the wildcards and /S direct BACKUP to examine every file on your disk. The /M switch directs BACKUP to only copy those files whose archive bit is set and to clear the archive bit after it successfully backs up the file to floppy disk. The /A switch directs BACKUP to add these files to the end of the previous day's incremental backup disk.

In this case, BACKUP will prompt you to insert the last incremental backup disk in drive A.

```
Insert last backup diskette in drive A:
Press any key to continue . . .
```

If this operation is your first incremental backup of the month, place a formatted disk in drive A and press ENTER. If you have performed previous incremental backups, place the last disk you used in drive A and press ENTER. BACKUP will add files to the disk until no more disk space remains. Should a disk fill, BACKUP will prompt you to insert a new disk. Place a formatted disk in drive A and press any key to continue. Label your incremental backup disk to avoid confusion, as shown in this example:



As you can see, by combining a complete system backup with daily or regular incremental backups, you keep your backups current. By the end of the month, you will have a large collection of disks. However, when you perform the next complete disk backup at the start of the following month, you can recycle the disks, repeating this process again. Using this method, you can limit the accumulation of backups to two months' worth.

Using BACKUP to Back Up a Key File or Directory

The previous backup commands assumed you want to back up your entire hard disk. There may be times, however, when you only want to back up one disk or directory. To do so, you invoke BACKUP specifying the directory name. For example, the following command backs up all the files in the DOS directory:

```
C:\> BACKUP  C:\DOS\*.*  A: <ENTER>
```

The following command backs up all the files in the DOS directory with the COM extension:

```
C:\> BACKUP  C:\DOS\*.COM  A: <ENTER>
```

Last, the following command backs up only the file FORMAT.COM from the DOS directory:

```
C:\> BACKUP  C:\DOS\FORMAT.COM  A: <ENTER>
```

As you can see, BACKUP lets you get very specific about the files you want to back up. If you don't have time to back up your entire disk, you should make time to back up your key files or directories.

Restoring Backup Files to Your Hard Disk

Hopefully you will never have to restore files from your backup floppies to your hard disk. Because most users seldom need their backup copies, they tend to pay less attention to performing regular backups. Unfortunately, almost like clockwork, when you quit performing regular file backups, either your disk will fail, or an inadvertent DOS command will destroy the files on your disk.

As the BACKUP command copies files from your hard disk to floppy, the RESTORE command copies backup files from floppy disks to your hard disk. Using RESTORE, you can copy every file on the backup disks or a specific file or directory back to your hard disk. To restore every file to your hard disk from the floppy disks in drive A, issue the following RESTORE command:

```
C:\> RESTORE  A:  C:\*.*  /S <ENTER>
```

Drive A, in this case, is the drive into which you will place the disks containing the files to restore. The wildcards direct RESTORE to restore every file on the floppy disk. As before, the /S switch directs RESTORE to restore files to every directory below the directory specified.

NOTE *The BACKUP command copies files to floppy disk using a unique storage format only it and RESTORE understand. Once you back up a file to floppy, the only way to place the backup copy back on to your disk is with RESTORE.*

When you issue the command above, RESTORE prompts you to insert the first backup disk into drive A.

```
Insert backup diskette 01 in drive A:
Press any key to continue . . .
```

After you place the disk in drive A and press ENTER, RESTORE will begin copying files from the backup disks to your hard disk, displaying the name of each file it successfully restores. If you are restoring your entire hard disk, begin with the first disk in your set of floppies from the complete disk backups. When RESTORE successfully restores all the files from the current floppy disk, it will prompt you to insert the next disk and press ENTER. After RESTORE successfully restores all the floppies, you must then repeat the RESTORE command to copy your incremental backups to the hard disk.

The two most common times you would perform a complete hard disk restore operation are when you have experienced a severe disk error losing all your files and when you are correcting disk fragmentation (the subject of Chapter 42). In most cases, you will use RESTORE to copy a specific file or directory of files as discussed next.

WARNING *RESTORE will overwrite existing files on your hard disk with the files on the backup disk. If you have changed a file's contents and have not yet backed up your changes, RESTORE might overwrite the current file with the file's previous contents.*

Restoring a Specific File or Directory

Unless your hard disk fails or you can't recover from a disk format operation, you will normally use the RESTORE command to restore a specific file or directory. For example, assume your disk contained the directory LETTERS, within which you stored your correspondence, and you inadvertently delete the files in the directory. If you are using DOS 5 or later or a third-party disk utility program, you may be able to *undelete* the files. If you don't have an undelete program available, or if your undelete utility is unsuccessful, you'll need to restore your backup copies of the files. In this case, you don't want to restore your entire disk, just the directory LETTERS. To begin, place your first backup disk from the set of complete disk backup floppies in drive A and issue the following RESTORE command:

```
C:\> RESTORE  A:  C:\LETTERS\*.* <ENTER>
```

RESTORE may have to examine several floppy disks before it locates the files to restore. To do so, RESTORE will prompt you to insert floppy disks into drive A as required. After RESTORE has examined your set of floppies from the complete disk backup, repeat the RESTORE command for your incremental floppies.

Although the process of examining every backup disk for matching files ensures RESTORE will locate the backup copies, you can easily see that looking at every backup disk could become quite time consuming. Luckily, if you are using DOS 3.3 or later, you can direct BACKUP to create a *log file* that lists which backup disks your files reside on. Using the backup log, you can quickly determine and select the disk containing the desired file or files and locate and restore the file much faster. If you are using DOS 3.3 or later, create a directory named BACKUP, in which you will store the backup log:

```
C:\> MKDIR  \BACKUP <ENTER>
```

Next, when you invoke BACKUP to perform a complete disk backup operation, include the /L switch directing BACKUP to use the file BACKUP.LOG. If a file named BACKUP.LOG currently exists in the BACKUP directory, delete or rename the file.

```
C:\> BACKUP  C:\*.*  A:  /S  /L:C:\BACKUP\BACKUP.LOG <ENTER>
```

Likewise, use /L when you perform an incremental backup.

```
C:\> BACKUP  C:\*.*  A:  /S  /A  /M  /L:C:\BACKUP\BACKUP.LOG
<ENTER>
```

Only delete or rename the BACKUP.LOG file before a complete disk backup. Your incremental backup operations will also log each filename in BACKUP.LOG as desired.

The backup log file contains each backup date and time, as well as the disk number containing each file, for example:

```
4-08-1993  18:27:04
001   \CONFIG.SYS
001   \AUTOEXEC.BAT
001   \NETWORK\NETWORK.BAT
001   \NETWORK\NET4.COM
001   \NETWORK\NET5.COM
001   \NETWORK\IPX.COM
002   \BATCH\DATETIME.BAT
002   \BATCH\CLEANUP.BAT
002   \BATCH\AUTOEXEC.SAV
002   \DOS\ANSI.SYS
```

Using the log file, you can quickly locate and restore key files.

Using Third-Party Backup Utility Programs

Many DOS users buy third-party backup utility programs that provide a menu-driven interface and improved performance. If you choose to buy and use third-party backup software, you will essentially still perform complete disk backups, as well as incremental disk backups. However, most third-party backup utilities let you select directories of files you don't want to back up because you can easily restore the corresponding files from your original disks. Whether you choose to use BACKUP and RESTORE or a third-party backup program, perform backup operations on a regular basis.

SUMMARY

Complete Disk Backup Operations

At the beginning of each month, you should back up every file on your hard disk to floppies. To do so, follow these steps:

1. Determine the number of floppy disks required using the CHKDSK command and Table 22-1.
2. Format the floppy disks using FORMAT.
3. Issue the following BACKUP command:

```
C:\> BACKUP \*.* A: /S <ENTER>
```

Incremental Backup Operations

Incremental backup operations direct BACKUP to copy only those files created or changed since the last complete backup operation. Because most users only create or change a few files each day, incremental backups take very little time. To keep your backups to date, perform the following BACKUP command on a regular basis:

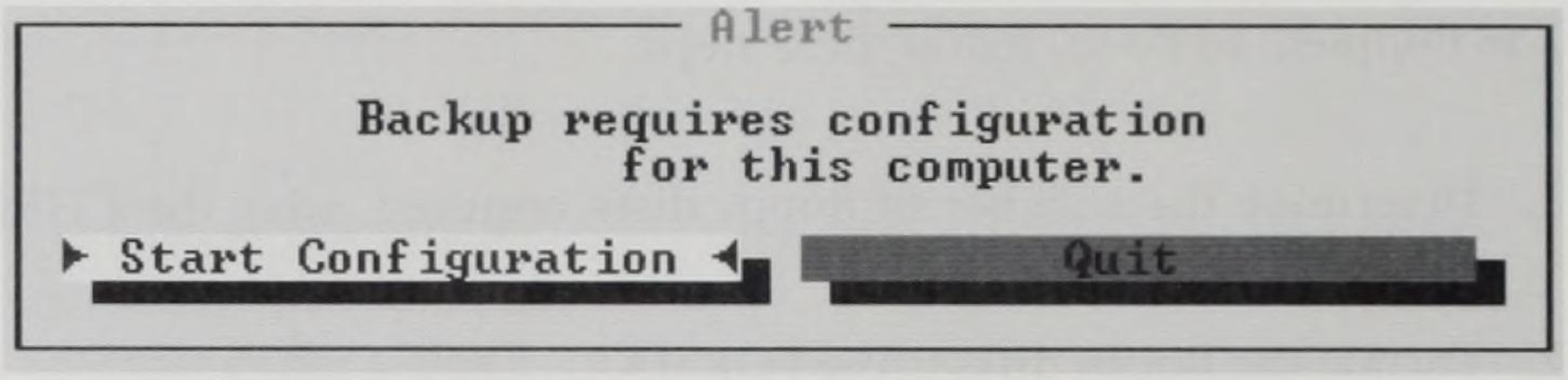
```
C:\> BACKUP C:\*.* A: /S /A /M <ENTER>
```

Using the DOS 6 MSBACKUP Command

If you are using DOS 6, the MSBACKUP command provides a menu-driven, fast backup utility. This section explains how to configure MSBACKUP and then how to use it to perform complete disk, as well as incremental and differential backup, operations.

Configuring MSBACKUP

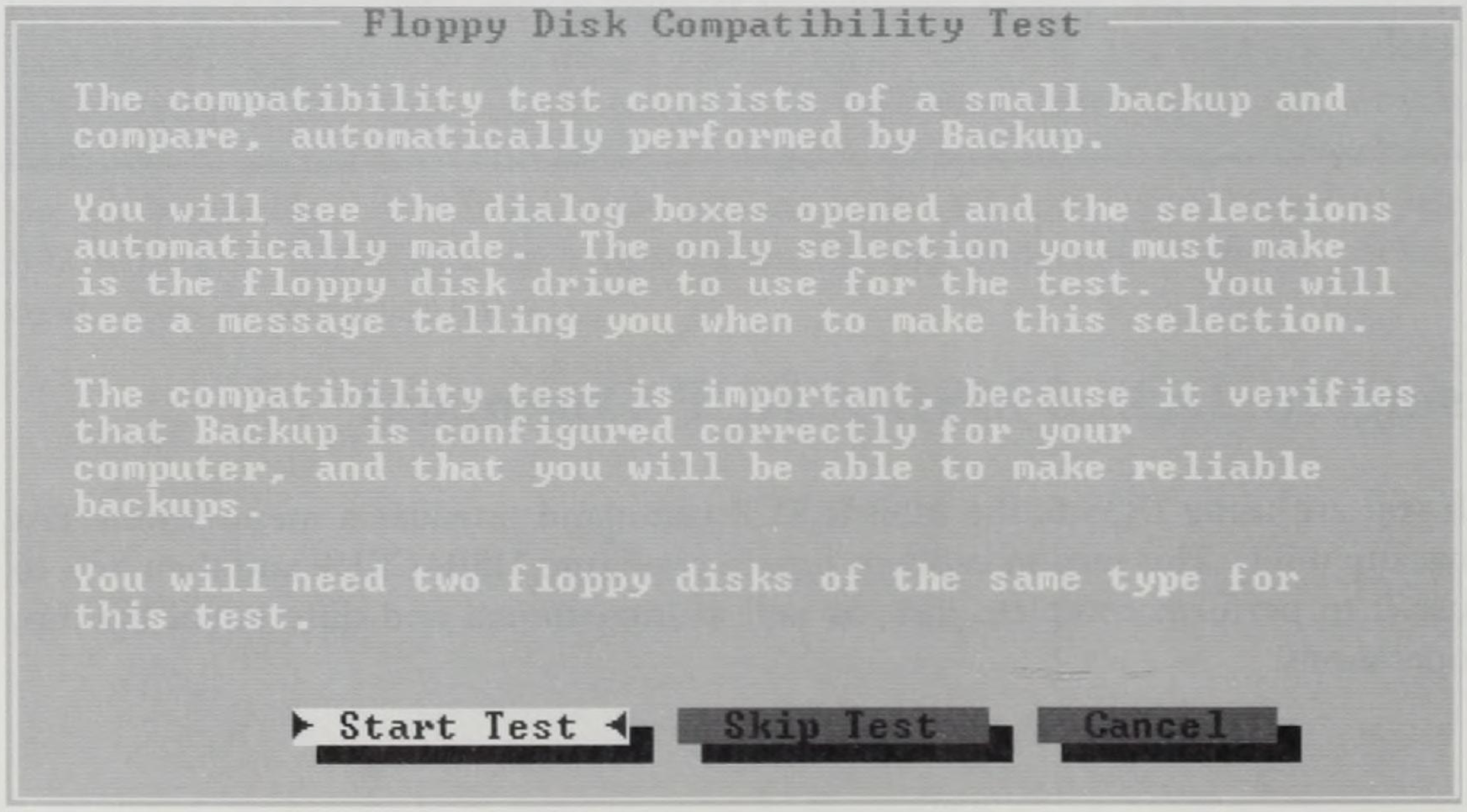
The first time you invoke MSBACKUP, it will display an Alert dialog box, as shown here, stating that you must configure the backup utility for your system.



Backup requires configuration
for this computer.

Select the Start Configuration option. MSBACKUP will first display its Video and Mouse Configuration dialog box. Select OK. MSBACKUP will then determine your system's floppy disk type, possibly testing your drive's change-line support (the ability to detect a floppy disk change). Follow the prompts displayed, generally selecting the OK option. MSBACKUP will eventually display its Floppy Disk Compatibility Test dialog box, as shown in Figure 22-1.

If you have two unused floppy disks readily available, perform the test by selecting the Start Test option. Otherwise, select Skip Test. MSBACKUP will display its Configure dialog box.



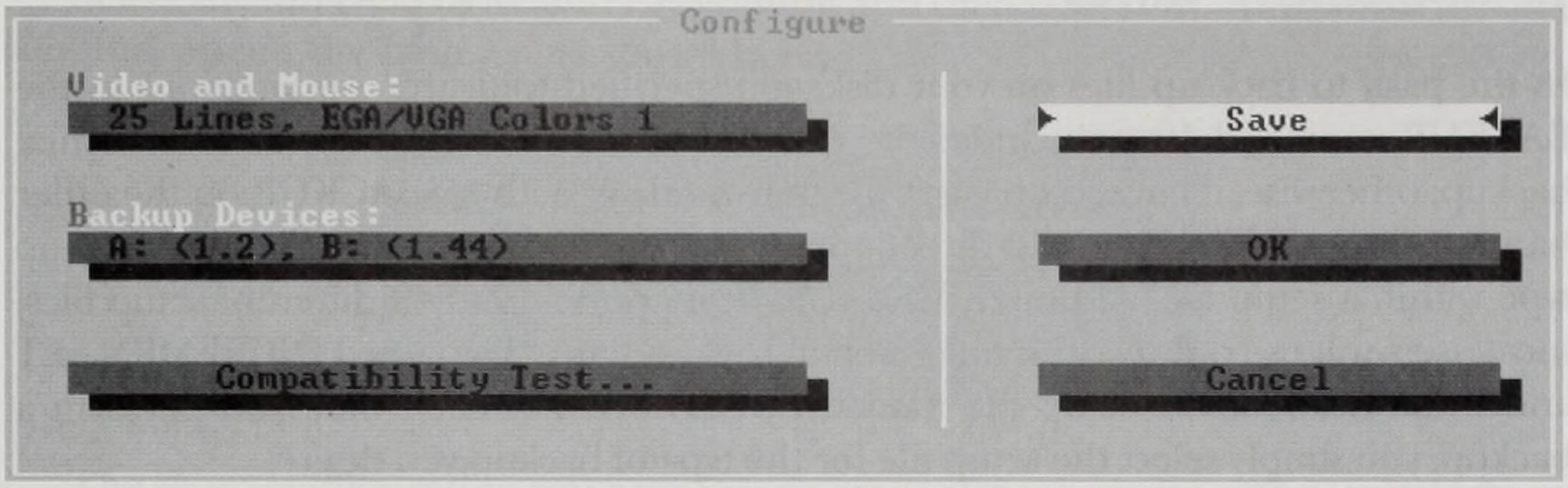
The compatibility test consists of a small backup and compare, automatically performed by Backup.

You will see the dialog boxes opened and the selections automatically made. The only selection you must make is the floppy disk drive to use for the test. You will see a message telling you when to make this selection.

The compatibility test is important, because it verifies that Backup is configured correctly for your computer, and that you will be able to make reliable backups.

You will need two floppy disks of the same type for this test.

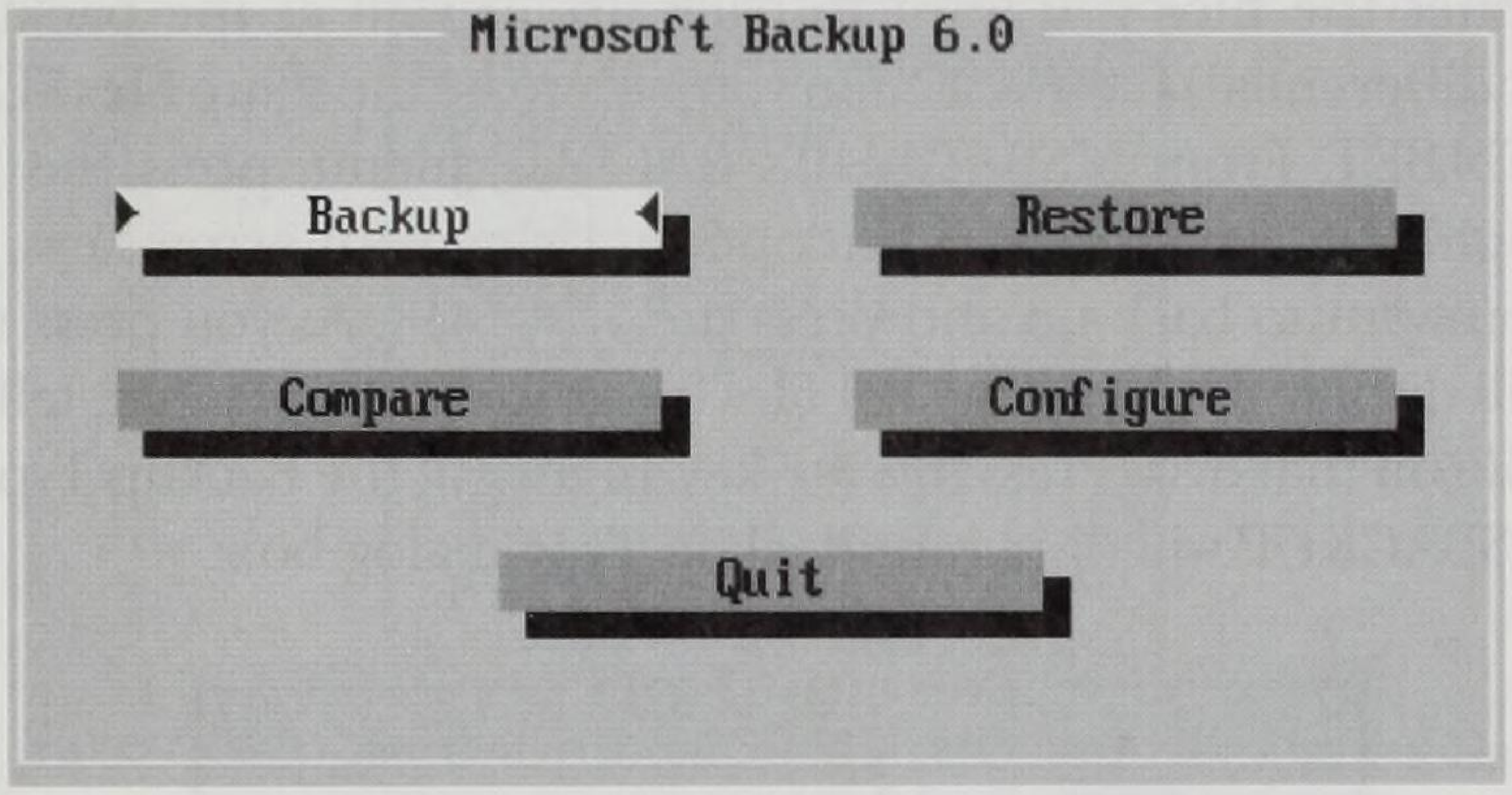
Figure 22-1. *The Floppy Disk Compatibility Test dialog box*



Video and Mouse:
25 Lines, EGA/VGA Colors 1

Backup Devices:
A: (1.2), B: (1.44)

Select the Save option. MSBACKUP will display its main menu, shown here. You are now ready to begin your backup operations.



Using MSBACKUP

MSBACKUP lets you perform three different types of backups: a *full* backup that backs up every file on your disk, an *incremental* (daily) backup that backs up every file you have created or changed since the last backup operation, or a *differential* backup that backs up the files you have created or changed since the last full backup operation. The difference between incremental and differential is that the incremental backs up those files created or changed since the last backup (any type of backup), whereas the differential backs up files created or changed since the last full backup. Although MSBACKUP supports several different command-line switches, most users will perform their backup and restore operations directly from MSBACKUP's main menu. As a result, invoke MSBACKUP from the DOS prompt as follows:

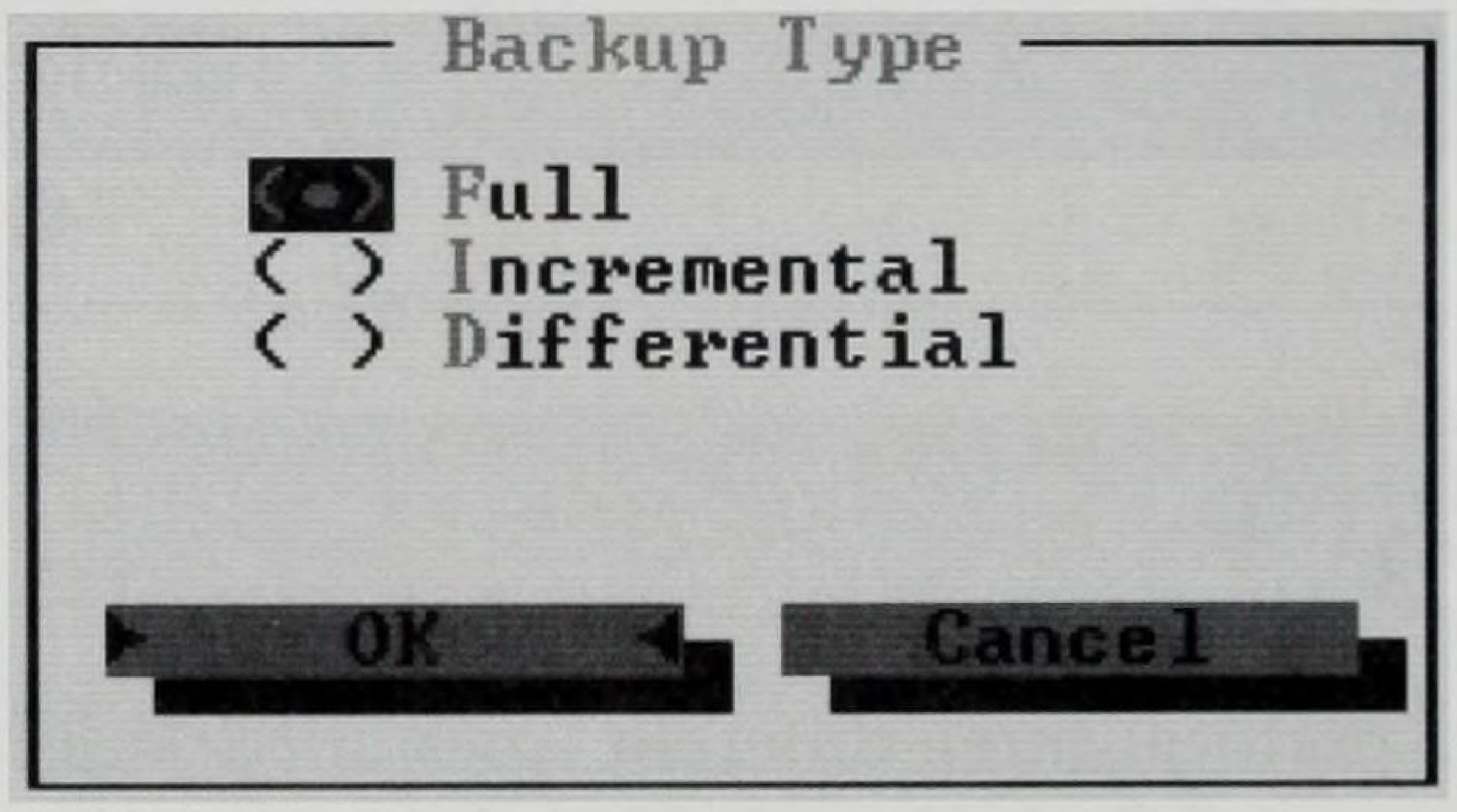
```
C:\> MSBACKUP <ENTER>
```

How MSBACKUP Works

In the past, to back up files on your disk, you specified wildcard combinations in the BACKUP command. If you included the /A switch, BACKUP performed an incremental backup; otherwise, it backed up every file on your disk. With MSBACKUP, on the other hand, you specify the files (and directions) you want to back up, as well as the backup type within a setup file. Although MSBACKUP supports up to 49 different setup files, most users will use only two: an incremental backup setup file named INCREMEN.SET and a full disk backup setup file named FULLDISK.SET. Each time you perform a backup, you simply select the setup file for the type of backup you desire.

Creating a Setup File

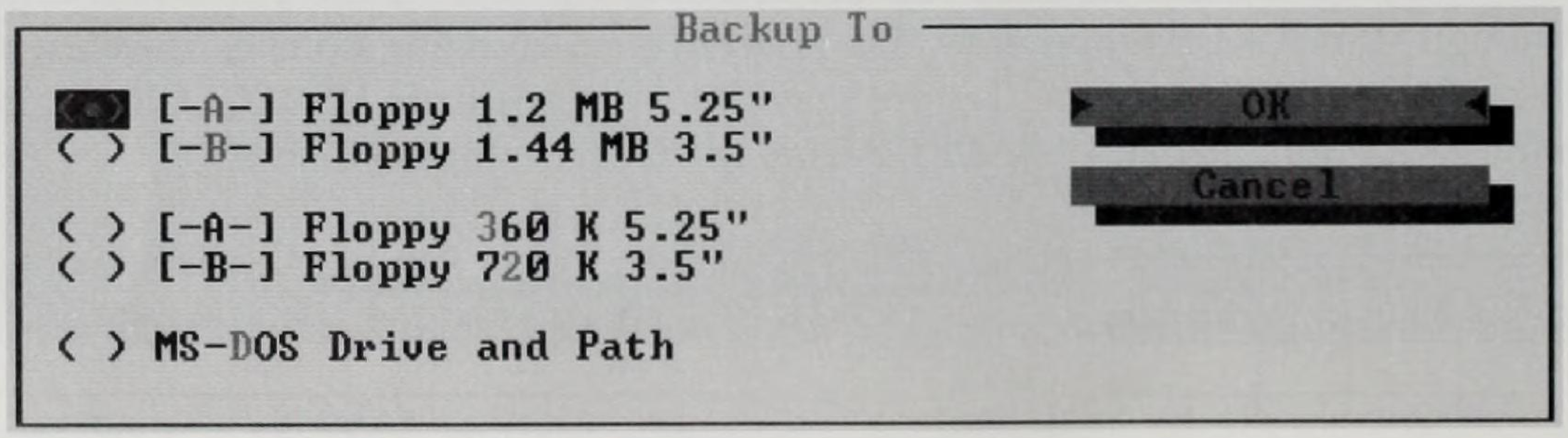
A *setup file* defines the files you want to back up, as well as the backup type (full, incremental, or differential). In this section you will create the setup files FULLDISK.SET and INCREMEN.SET. From MSBACKUP's BACKUP menu, press the TAB key until MSBACKUP highlights the Backup From option. Using your arrow keys, highlight the disk or disks you want to back up and press the SPACEBAR. As you press the SPACEBAR, MSBACKUP will display the message "All files" next to the drive letter, telling you it will back up every file on that disk. Press the TAB key, highlight the Backup Type option, and press ENTER. MSBACKUP will display its Backup Type dialog box:



Backup Type

● Full
○ Incremental
○ Differential

Use your keyboard arrow keys to highlight the desired backup type, in this case "Full," and press the SPACEBAR, followed by ENTER. Press the TAB key to highlight the Backup To option. If it does not currently say MS-DOS Drive and Path, press ENTER. MSBACKUP will display its Backup To dialog box:



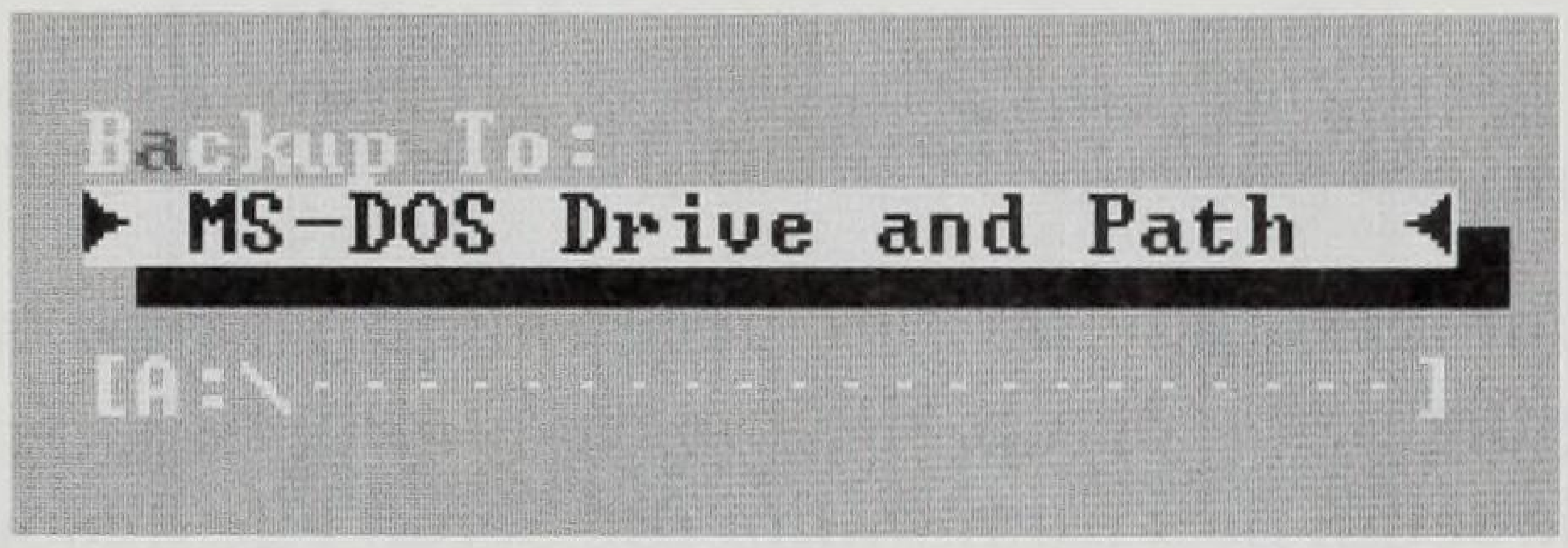
Backup To

● [-A-] Floppy 1.2 MB 5.25"
○ [-B-] Floppy 1.44 MB 3.5"

○ [-A-] Floppy 360 K 5.25"
○ [-B-] Floppy 720 K 3.5"

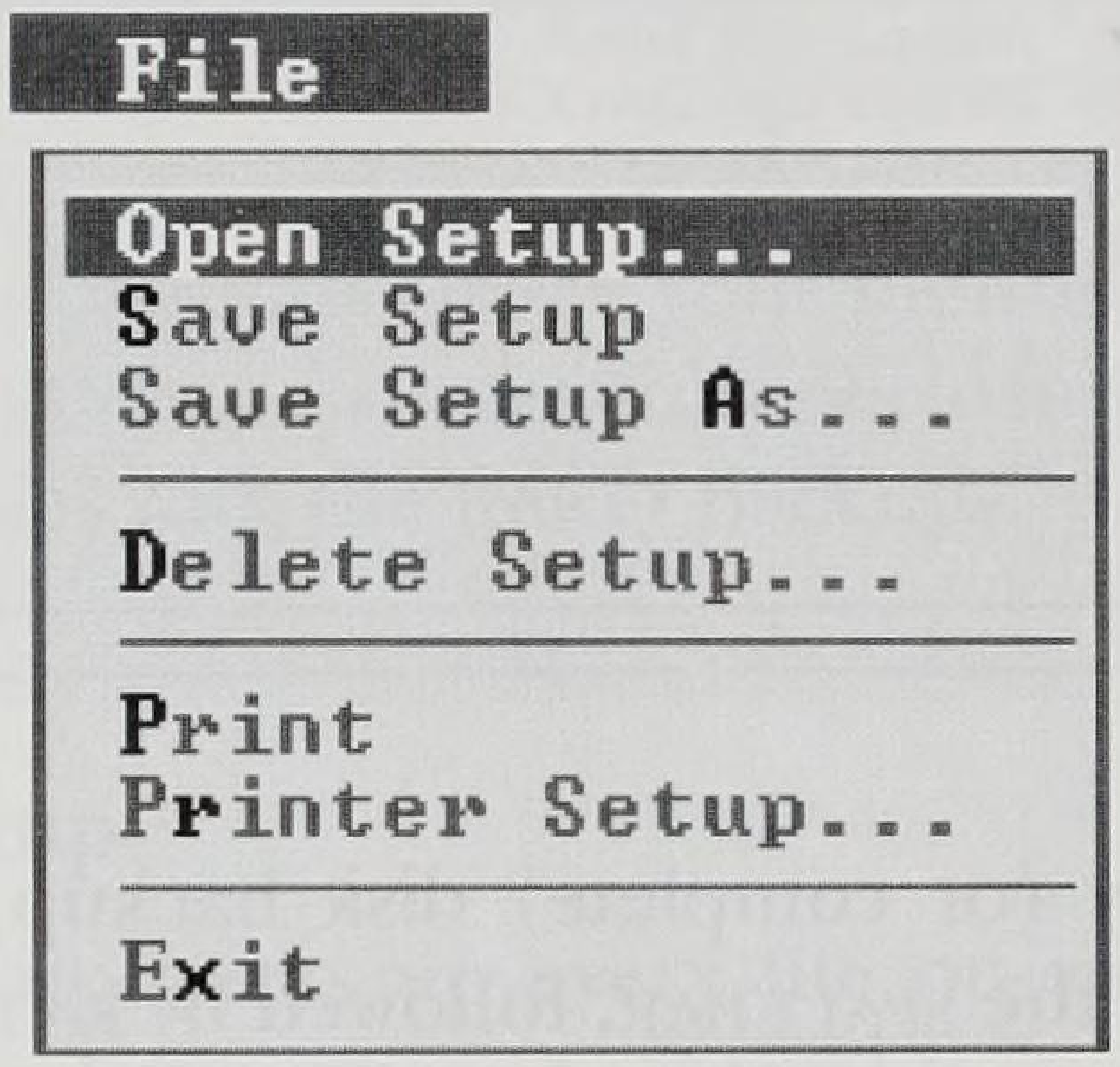
○ MS-DOS Drive and Path

Select the MS-DOS Drive and Path option and then select OK. Next, press the TAB key and type in the path **A:\**, as shown here:

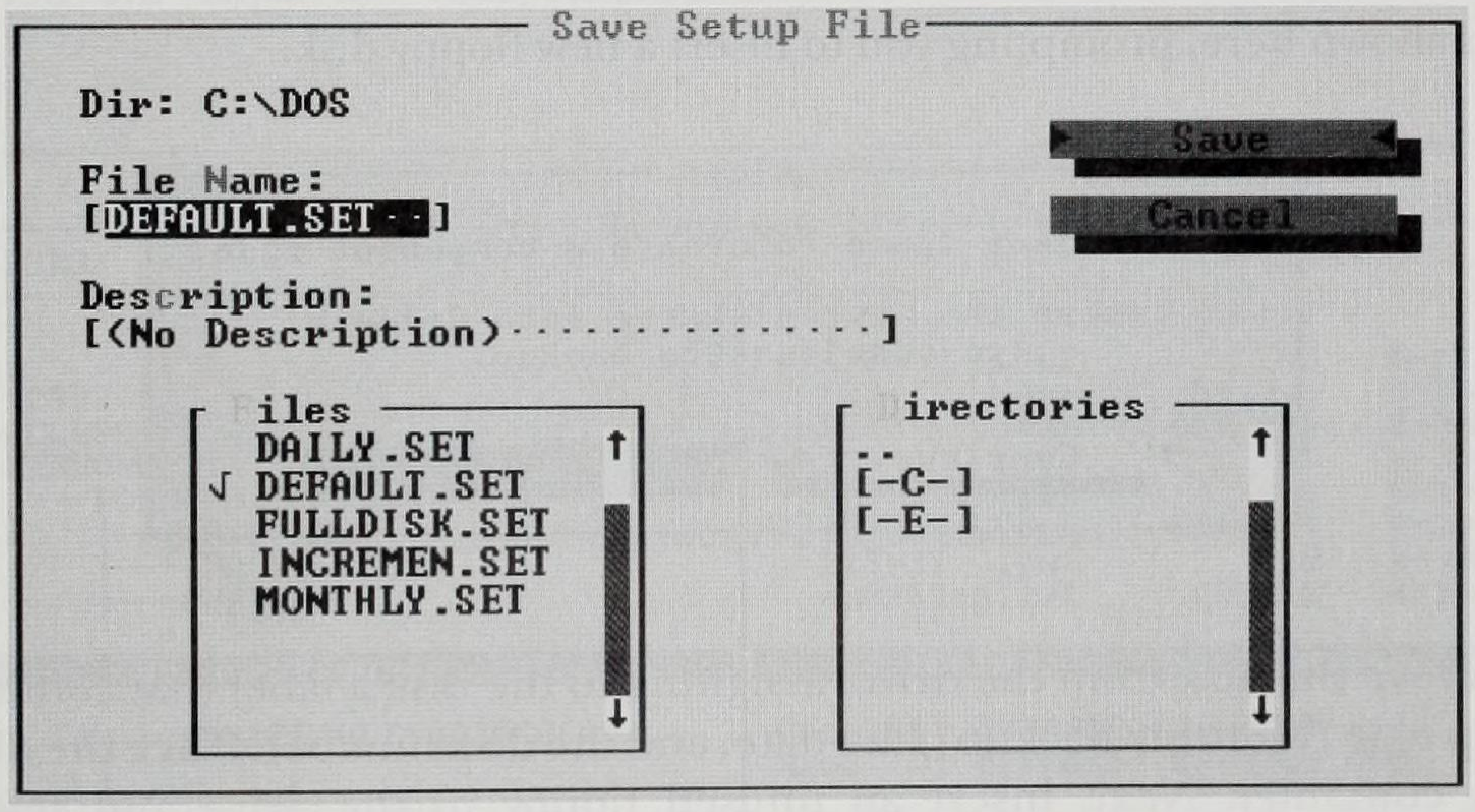


Backup To:
▶ MS-DOS Drive and Path ◀
[A:\······················]

The setup configuration you have just created will back up every file on disk. Press the ALT-F keyboard combination to select the File menu:



Select the Save Setup As option. MSBACKUP will display its Save Setup File dialog box, as shown in the following.

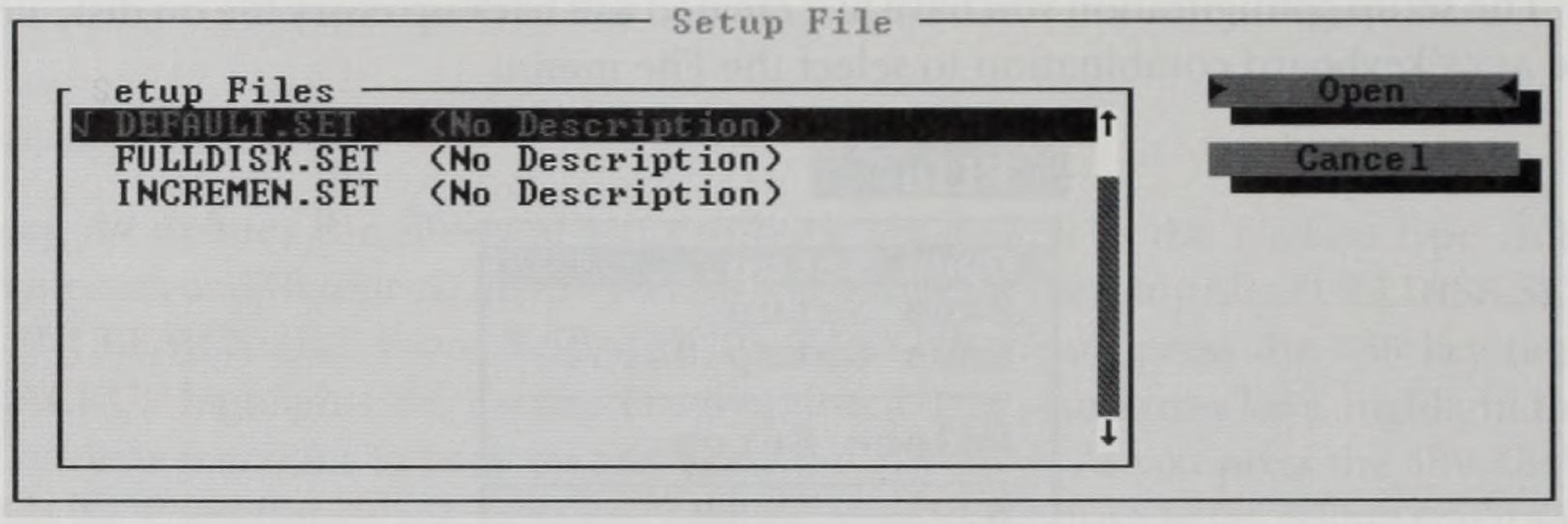


Save Setup File
Dir: C:\DOS
File Name:
[DEFAULT.SET··]
Description:
[(No Description)··············]
Files
DAILY.SET
√ DEFAULT.SET
FULLDISK.SET
INCREMEN.SET
MONTHLY.SET
Directories
..
[-C-]
[-E-]

Type in the filename **FULLDISK.SET** and press ENTER. MSBACKUP will save the configuration to the file. Next, press the TAB key and highlight the Backup Type option, again pressing ENTER. When MSBACKUP displays the Backup Type dialog box, select the Incremental option, followed by OK. Next press the ALT-F keyboard combination to select the File menu. As before, choose the Save Setup As option and type in the filename INCREMEN.SET. Your two backup setup files are now ready for use.

Performing a Monthly Backup

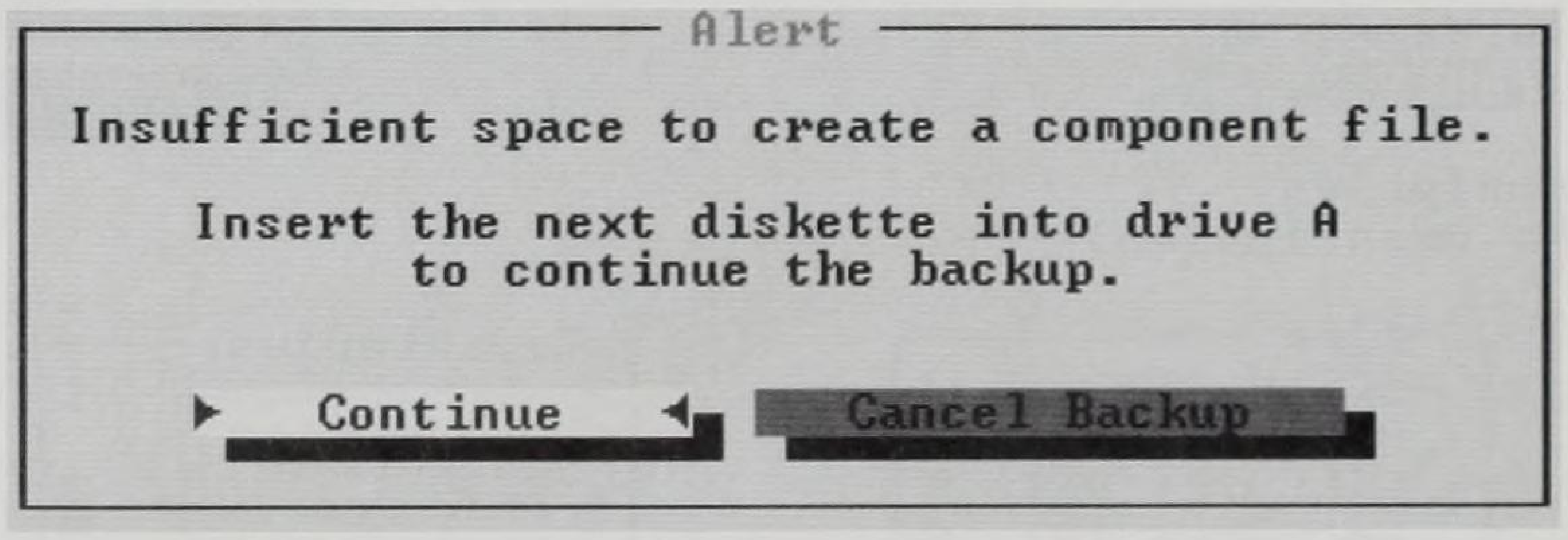
After you define your backup setup files, performing backups is very easy. From the Backup menu, highlight the Setup File option and press ENTER. MSBACKUP will display the available setup files from within the Setup File dialog box:



```
Setup File
etup Files
√ DEFAULT.SET   (No Description)
  FULLDISK.SET  (No Description)
  INCREMEN.SET  (No Description)
```

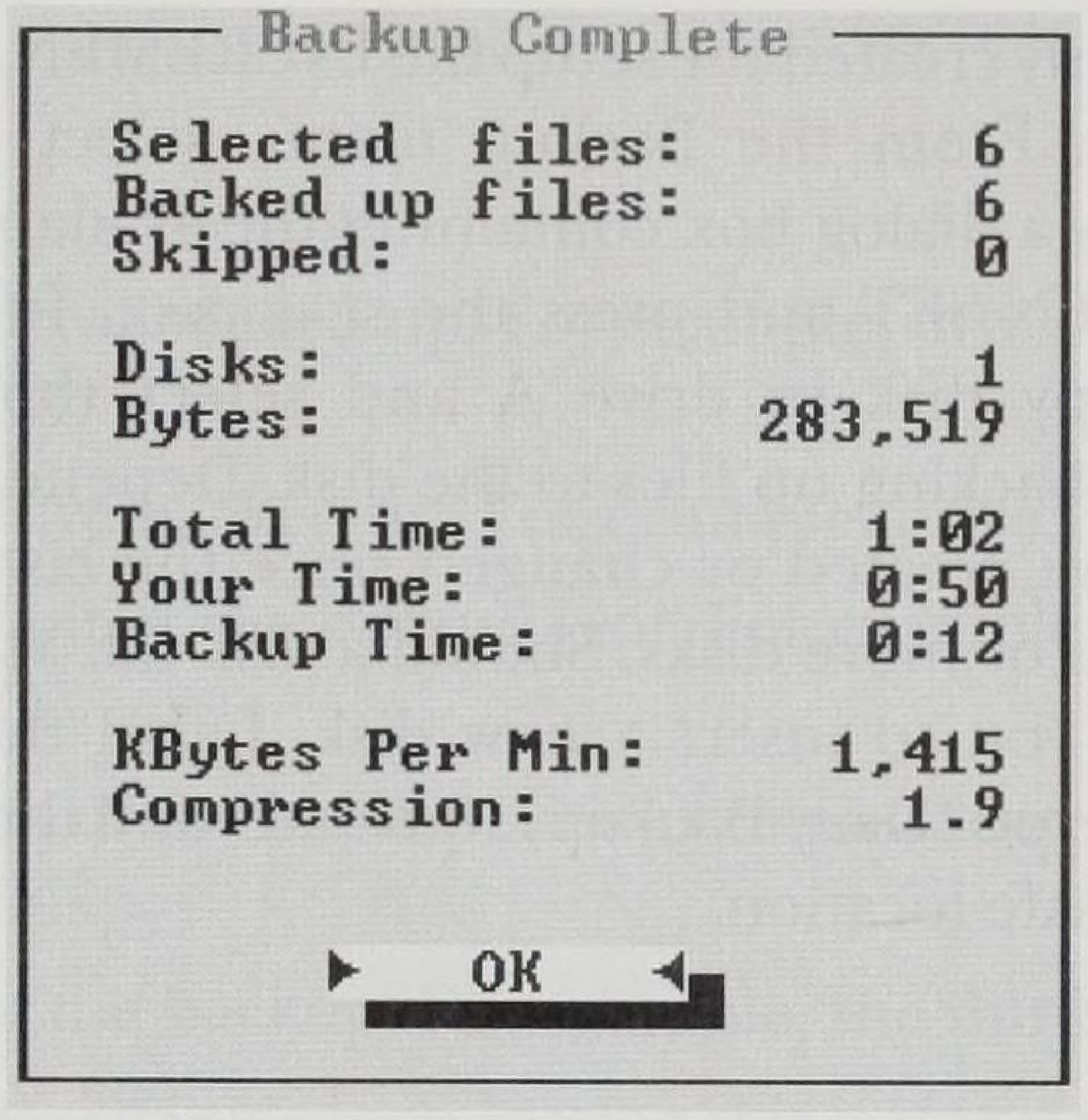
To perform a monthly (or complete) disk backup, highlight the file named FULLDISK.SET and press the SPACEBAR, followed by ENTER. Next, place an unused, formatted floppy disk in drive A. Press the TAB key to highlight the Start Backup option. As the backup begins, MSBACKUP will display a screen summarizing the operation, as shown in Figure 22-2.

Eventually, the disk in drive A will fill and MSBACKUP will display the Alert dialog box, as shown here, prompting you to insert a new floppy disk.



```
Alert
Insufficient space to create a component file.

Insert the next diskette into drive A
to continue the backup.
```

Remove the disk from the drive, assigning to the disk a label that contains the backup type (Monthly Backup), the date, and the disk number. Place the disk into a disk container. Next, insert an unused floppy in the drive and select the Continue option. Repeat this process of inserting and labeling floppy disks until the backup completes. When the backup ends, MSBACKUP will display the Backup Complete dialog box.

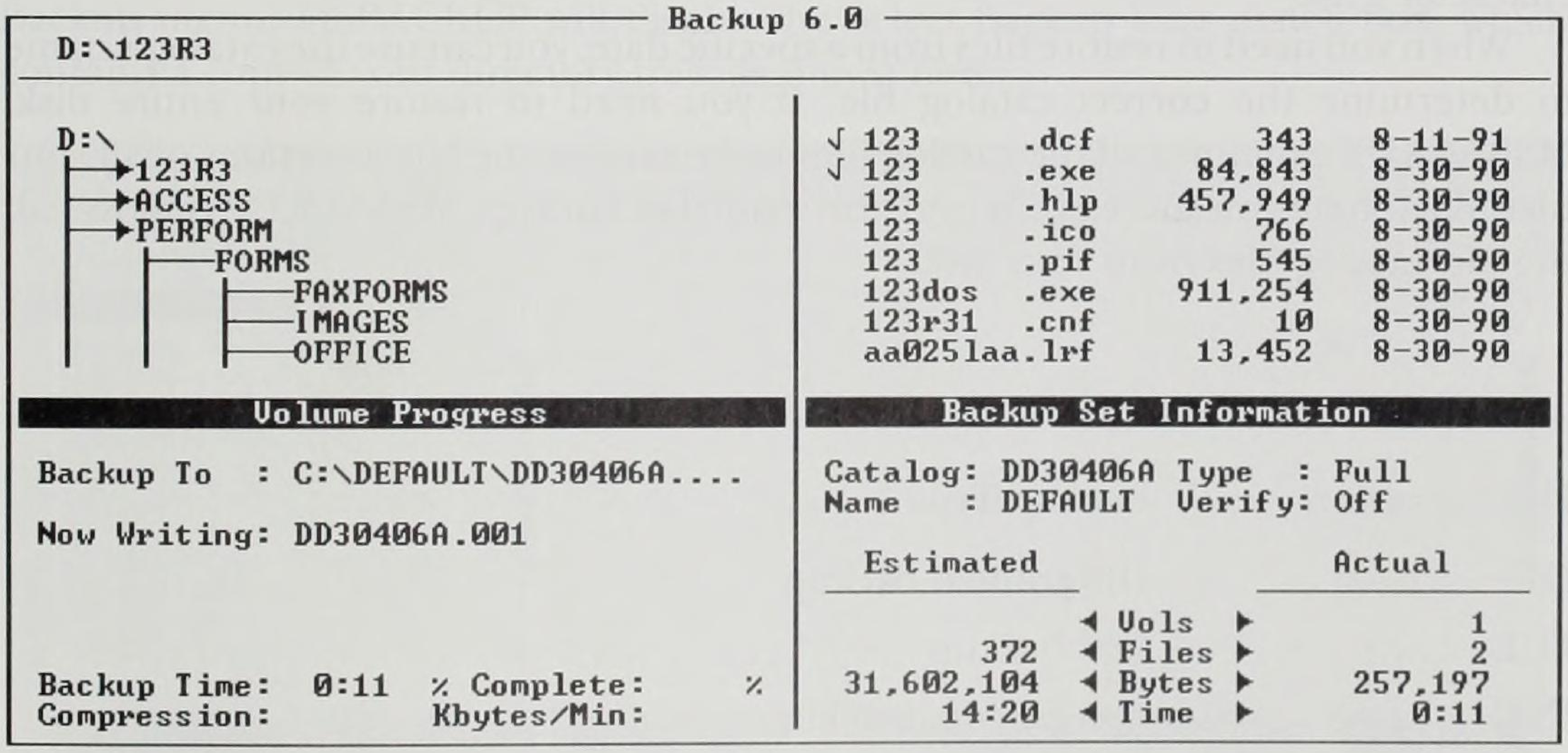


```
Backup Complete

Selected files:          6
Backed up files:         6
Skipped:                 0

Disks:                   1
Bytes:             283,519

Total Time:           1:02
Your Time:            0:50
Backup Time:          0:12

KBytes Per Min:      1,415
Compression:           1.9

          ► OK ◄
```

Select the OK option to continue. Place your monthly backup floppy disks in a safe location. Next month, after you successfully perform a full disk backup, you can replace these backup floppies with the newer backups.

Performing a Daily Backup

Because the monthly backup backs up every file on your disk, it can be quite a time-consuming process and use up a considerable number of floppies. Luckily, however, you only have to perform the operation once a month. The remainder of the month, you can perform a daily or incremental backup. The incremental backup backs up only those files you have created or changed since the last backup operation.



```
                         Backup 6.0
D:\123R3

D:\                               √ 123      .dcf        343  8-11-91
 ├──►123R3                        √ 123      .exe     84,843  8-30-90
 ├──►ACCESS                         123      .hlp    457,949  8-30-90
 ├──►PERFORM                        123      .ico        766  8-30-90
 │    ├────FORMS                    123      .pif        545  8-30-90
 │    │     ├────FAXFORMS           123dos   .exe    911,254  8-30-90
 │    │     ├────IMAGES             123r31   .cnf         10  8-30-90
 │    │     ├────OFFICE             aa025laa.lrf      13,452  8-30-90

         Volume Progress                  Backup Set Information

Backup To   : C:\DEFAULT\DD30406A....   Catalog: DD30406A Type   : Full
                                        Name   : DEFAULT  Verify: Off
Now Writing: DD30406A.001
                                          Estimated             Actual

                                                    ◄ Vols  ►          1
                                               372  ◄ Files ►          2
Backup Time:  0:11  % Complete:     %   31,602,104  ◄ Bytes ►    257,197
Compression:        Kbytes/Min:              14:20  ◄ Time  ►       0:11
```

Figure 22-2. *The Backup status screen*

Because you previously created the setup file INCREMEN.SET, performing a daily backup is very simple. From the Backup menu, select the Setup File option. MSBACKUP will display a dialog box containing the available setup files. Highlight the filename INCREMEN.SET and press the SPACEBAR, followed by ENTER. Next, place an unused floppy disk in drive A and select the Start Backup option. MSBACKUP will begin backing up files to the disk. Depending on the number and size of the files you have created or changed since the last backup operation, the backup may require two or more disks. Should a disk fill, MSBACKUP will display a dialog box prompting you to insert a new disk. Label the old disk with a label containing the backup type (Daily Backup), the date, and the disk number. Place the daily backup disks in a safe location.

Understanding Backup Sets

To simplify the process of restoring files, the MSBACKUP command supports *backup sets* and *catalogs*. A backup set has a catalog that describes the following:

- The number of files backed up
- Specifics about each file backed up
- The MSBACKUP setup file used to perform the backup
- The date of the backup

MSBACKUP stores the catalog files in the DOS directory. Depending on the backup type the files will have one of the extensions specified in Table 22-2.

The catalog file's eight-character name tells you specifics about the catalog. Each character in the filename has a specific meaning. Table 22-3 briefly describes each character's use.

When you need to restore files from a specific date, you can use the catalog's name to determine the correct catalog file. If you need to restore your entire disk, MSBACKUP examines all the catalog files to determine the latest versions of all your files. Each time you successfully perform a full disk backup, MSBACKUP removes all the old catalog files from your disk.

File Extension	Backup Type
DIF	Differential backup
FUL	Full backup
INC	Incremental backup

Table 22-2. MSBACKUP Catalog Extensions

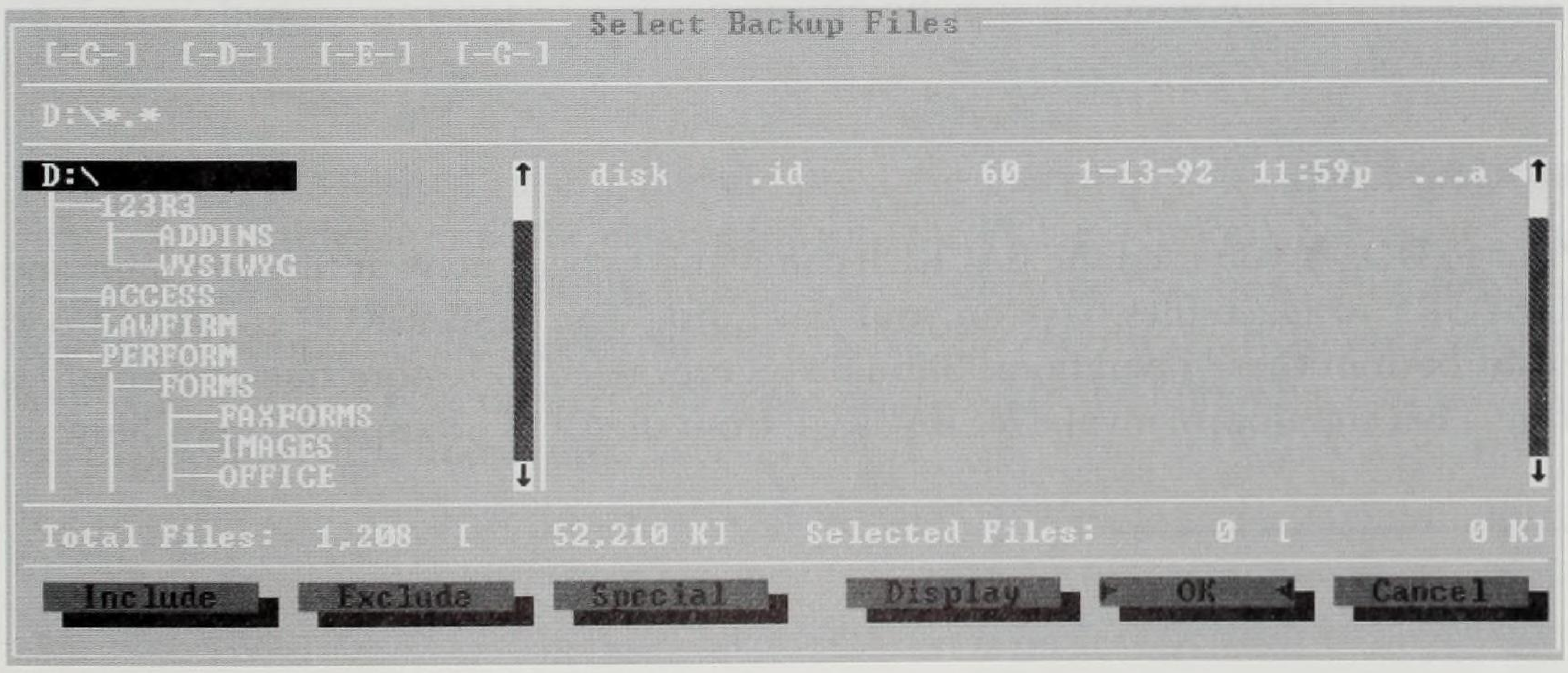
Character Position(s)	Meaning
1	The first drive backed up. For example, drive C would be C*XXXXXXX*.
2	The last drive backed up. For example, drive E would be *X*E*XXXXXX*.
3	The last digit of the backup year. For example, 1993 would be *XX*3*XXXXX*.
4, 5	The month of the backup. For example, September would be *XXX*09*XXX*.
6, 7	The day of the backup. For example, the third would be *XXXXX*03*X*.
8	A letter describing which catalog this is if you performed several backups on the same day. The first catalog gets the letter A, the second B, and so on.

Table 22-3. The Meaning of Letters Within a MSBACKUP Catalog's Filename

Selecting and Excluding Specific Files

The previous backup operations assumed that you wanted to back up every file on your disk. Most users, however, have many different application programs on their disk such as Windows, Excel, or Word. Because they can easily install these programs from the original floppies, backing up the files from their hard disk simply consumes time and floppies. One of the most powerful features MSBACKUP provides is the ability to select or exclude specific files for backing up.

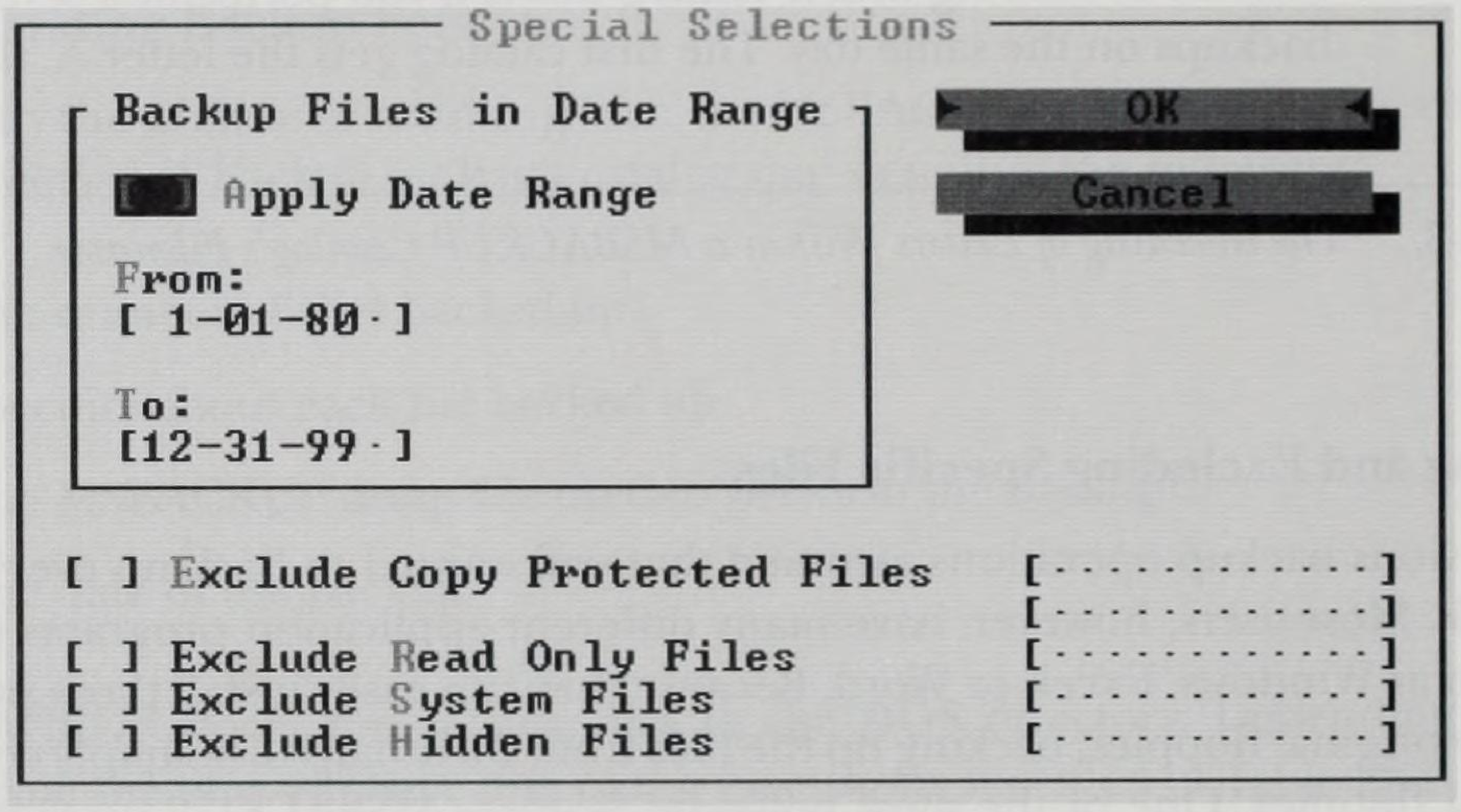
To exclude or select specific directories, choose the Select Files option from the Backup menu. MSBACKUP will display the Select Backup Files dialog box, which contains a copy of your directory tree, as shown here:



```
D:\*.*

D:\                                  disk     .id        60   1-13-92  11:59p  ...a
 ├─123R3
 │  ├─ADDINS
 │  └─WYSIWYG
 ├─ACCESS
 ├─LAWFIRM
 ├─PERFORM
 │  ├─FORMS
 │  │  ├─FAXFORMS
 │  │  ├─IMAGES
 │  │  ├─OFFICE

Total Files:  1,208  [    52,210 K]   Selected Files:        0  [          0 K]
```

Using your keyboard arrow keys, highlight the directories you want to back up and press the SPACEBAR. MSBACKUP will display a triangle next to the directory name, indicating it will back up the directory's files. To exclude a directory, don't press the SPACEBAR to select it, or if the directory is already selected, press the SPACEBAR a second time, deselecting it (MSBACKUP will change the triangle's appearance indicating the exclusion). By pressing the TAB key you can also move the cursor to the list of files contained in a directory and individually select or deselect files using this same technique. After you select the last files you desire, select the OK option. Use the File menu Save option to save the selected directories within the desired setup file.

There may be times when you want to back up files that were created or changed after or before a specific date. To do so, select the Special option from within the Select Backup Files dialog box. MSBACKUP will display the Special Selections dialog box.



Special Selections

Backup Files in Date Range

[ ] Apply Date Range

From:
[1-01-80]

To:
[12-31-99]

[ ] Exclude Copy Protected Files [ ]
[ ]

[ ] Exclude Read Only Files [ ]
[ ] Exclude System Files [ ]
[ ] Exclude Hidden Files [ ]

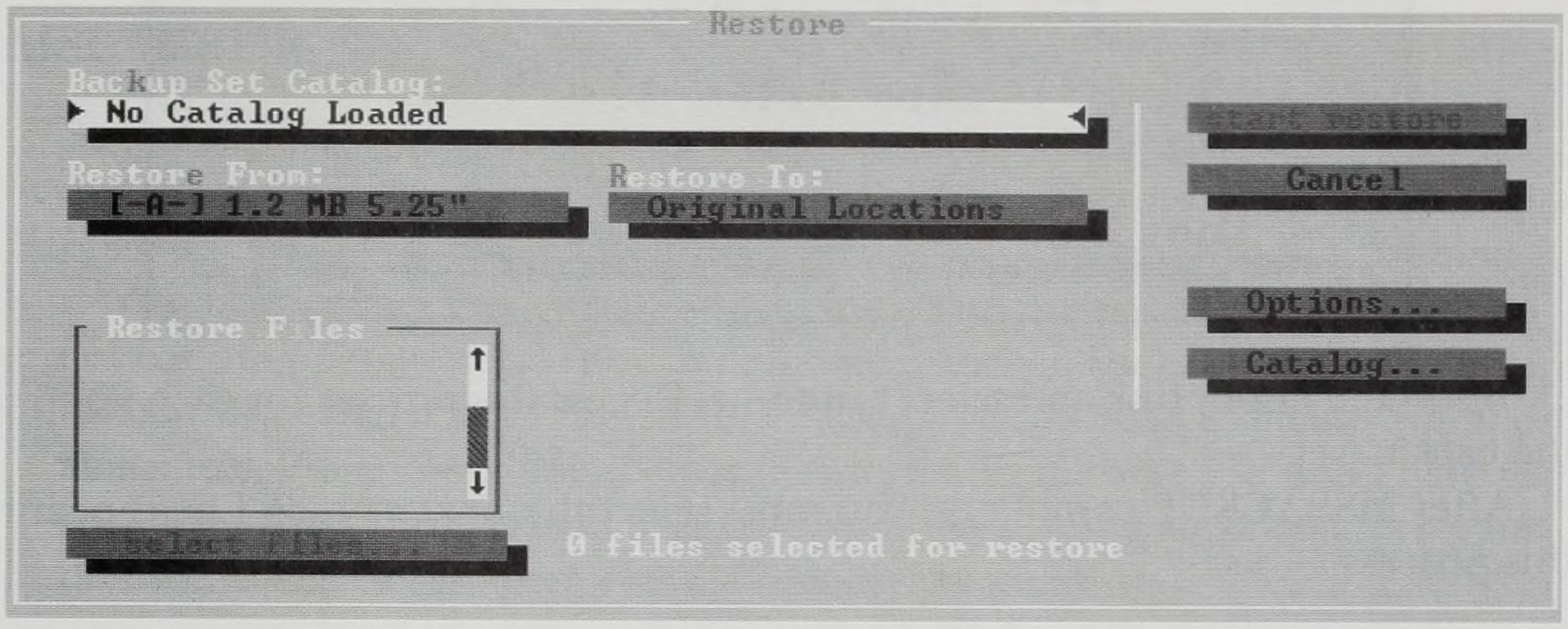
To select specific dates, highlight the Apply Date Range option and press the SPACEBAR (placing a check mark in the box). Next, type in the desired dates in the From and To fields. By placing check marks in the remaining boxes, MSBACKUP lets you exclude files based on other attribute settings. In general, most users will only exclude files by not selecting specific directories, as previously discussed.

Restoring Files

Prior to DOS 6, you used the BACKUP command to back up your files to floppy and RESTORE to place files back on your hard disk. The MSBACKUP command performs both of these operations. Should you ever need to restore one or more files from a backup floppy, invoke MSBACKUP from the DOS prompt as shown here:

```
C:\> MSBACKUP <ENTER>
```

When MSBACKUP displays its main menu, select the Restore option. MSBACKUP will display the Restore screen, as shown in the following.



Backup Set Catalog: ▶ No Catalog Loaded

Restore From: [-A-] 1.2 MB 5.25"

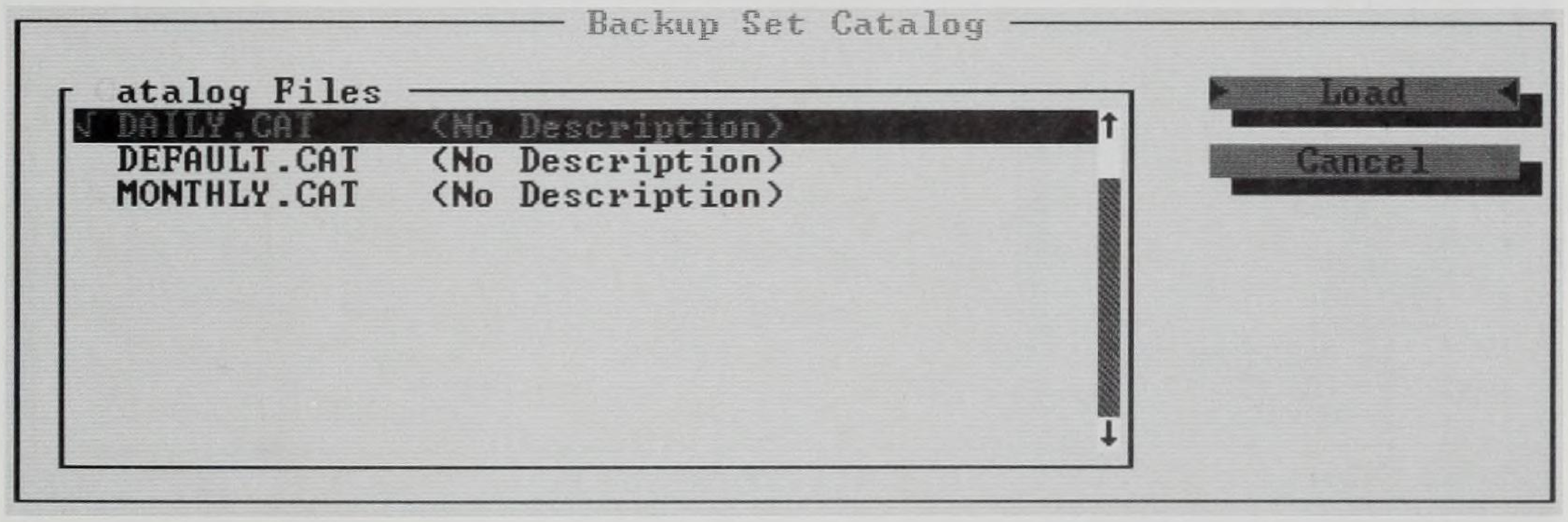
Restore To: Original Locations

0 files selected for restore

Performing a Complete Disk Restore

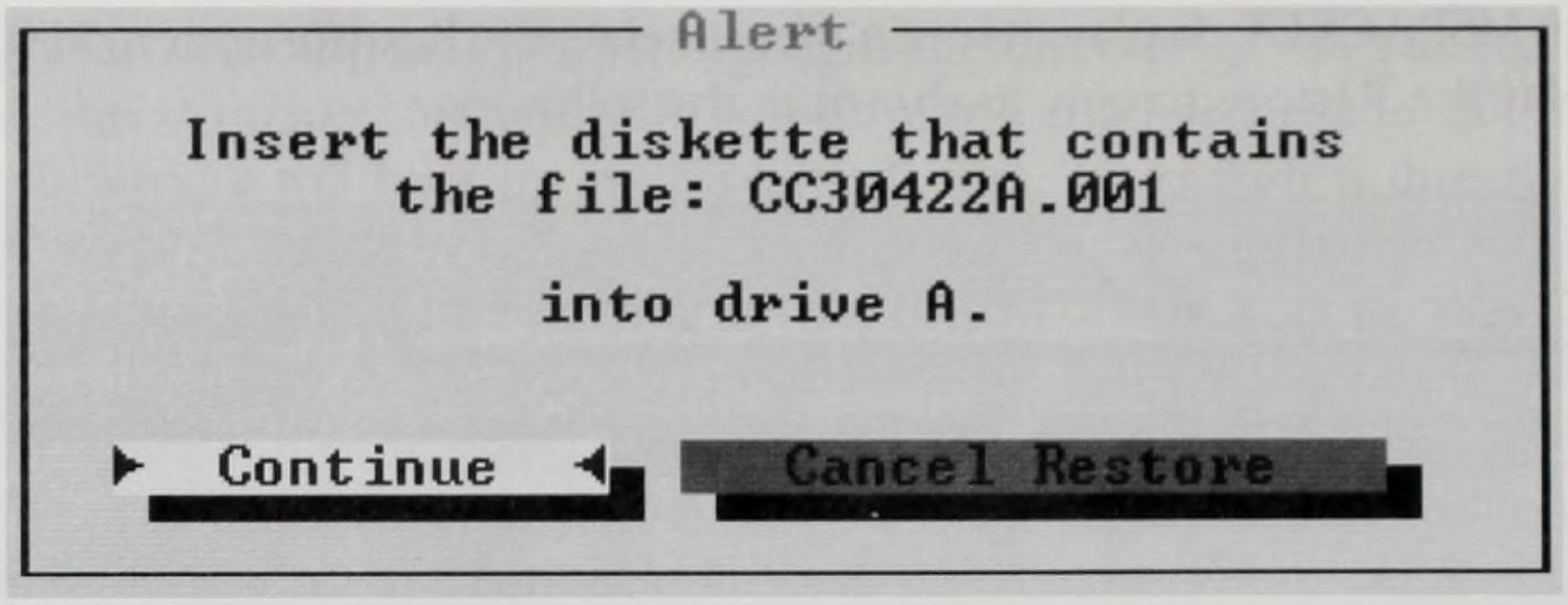
There may be times when you need to restore every file on your backup disks.

To begin, press the TAB key to highlight the Backup Set Catalog option and press ENTER. MSBACKUP will display the Backup Set Catalog dialog box, shown here, which contains the available catalog files:



```
atalog Files
√ DAILY.CAT      <No Description>
  DEFAULT.CAT    <No Description>
  MONTHLY.CAT    <No Description>
```

Highlight the desired catalog file and press the SPACEBAR. You will want to begin with your monthly backup (the catalog file with the FUL extension). Select the Load option. Make sure the Restore To option is set to Original Locations. Highlight the Restore Files option and select All Files. Next, choose the Start Restore option. MSBACKUP will display an Alert dialog box, as shown in the following, that prompts you to insert the first floppy disk.



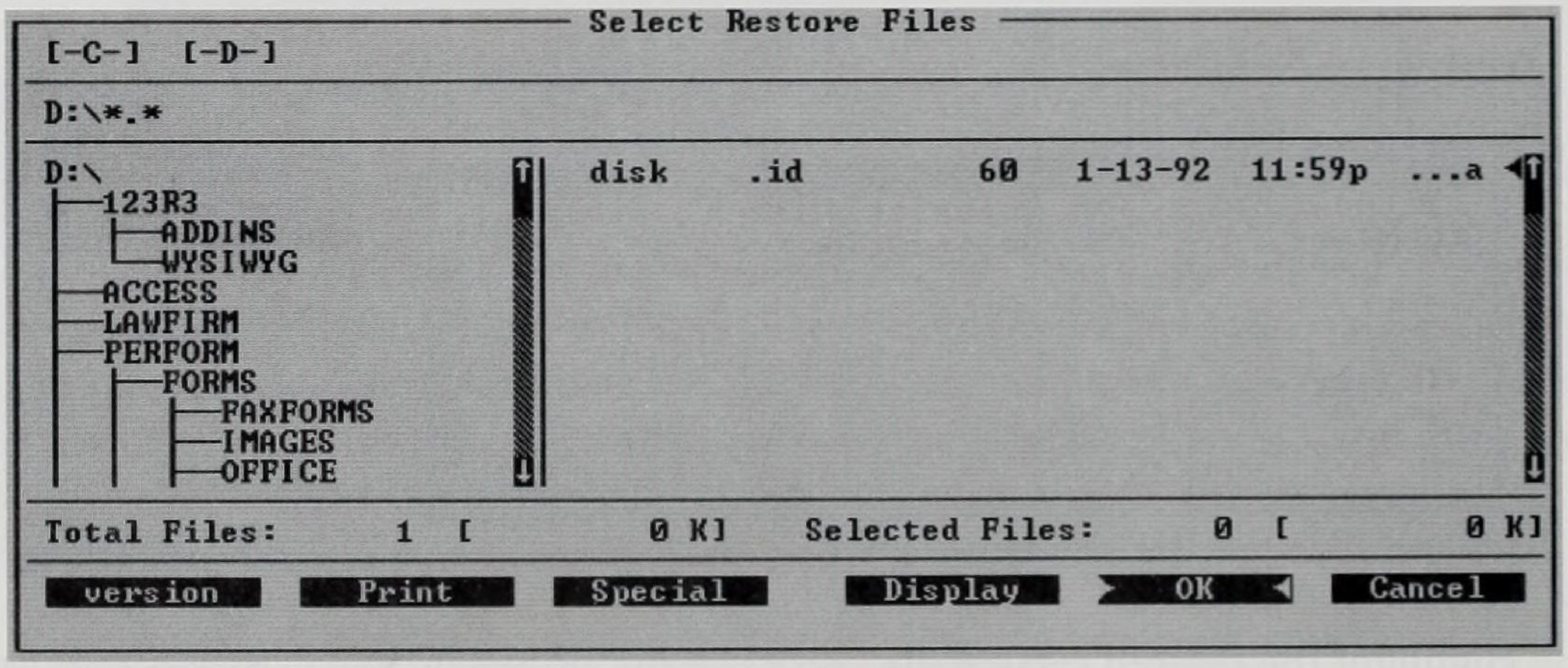
```
Insert the diskette that contains
the file: CC30422A.001

into drive A.
```

Insert the floppy disk and select Continue. Repeat this process for each floppy in the backup set.

After MSBACKUP completes, you must select the next catalog file, repeating this process.

Restoring Specific Files or Directories

The steps you need to restore a specific directory or files is very similar to those you performed for a disk backup operation. To begin, select the desired catalog file. When MSBACKUP displays the Restore screen, choose the Select Files option. MSBACKUP will display the Select Restore Files dialog box, shown here, which contains a directory tree of files available for restoration:



```
D:\*.*

D:\                      disk    .id        60   1-13-92  11:59p  ...a
├─123R3
│ ├─ADDINS
│ └─WYSIWYG
├─ACCESS
├─LAWFIRM
├─PERFORM
│ ├─FORMS
│ │ ├─FAXFORMS
│ │ ├─IMAGES
│ │ ├─OFFICE

Total Files:     1 [       0 K]   Selected Files:      0 [          0 K]
```

Using your keyboard arrow keys, highlight the directory containing the files you want to restore and press the SPACEBAR. MSBACKUP will display a triangle next to the directory name indicating its selection. Repeat this process for each directory you want to restore. Select the OK option. Next, select Start Restore. As before, MSBACKUP will display the Alert dialog box asking you to insert the first backup floppy. Insert the disk and select Continue. Repeat this process for each floppy disk in the backup set.

After the restore operation completes, select additional catalog files performing this same processing.

SUMMARY

MSBACKUP

Beginning with version 6, DOS provides the MSBACKUP command, a menu-driven utility that lets you easily perform full, incremental, or differential backup operations. A full backup operation backs up every file on your disk. An incremental backup backs up only those files you have created or changed since the last backup operation. Likewise, a differential backup backs up those files you created or changed since the last full backup. MSBACKUP also gives you the ability to select and exclude specific directories. In this way, you don't have to spend time backing up program files you could install using the original program disks. MSBACKUP also performs file restore operations.

Hands On

If you have never performed a complete disk backup or have not done so in some time, use BACKUP to do so now. To begin, use the CHKDSK command as discussed to determine the amount of disk space currently in use.

```
C:\> CHKDSK <ENTER>
```

Using Table 22-1, determine how many floppy disks you will need to back up your disk. Next, format the number of disks specified. Use the following BACKUP command to back up all the files on your disk:

```
C:\> BACKUP  C:\*.*  A:  /S <ENTER>
```

Correctly label each disk and place the disks into a disk storage container.

The best way to understand the BACKUP and RESTORE process is to try it. To begin, create the directory TESTINFO on your hard disk.

```
C:\> MKDIR  \TESTINFO <ENTER>
```

Next, copy all files in your DOS directory with the SYS extension to this directory.

```
C:\> COPY  \DOS\*.SYS  \TESTINFO <ENTER>
```

Next, back up the files in the TESTINFO directory to drive A.

```
C:\> BACKUP  C:\TESTINFO\*.*  A: <ENTER>
```

When the BACKUP command completes, delete the files in the TESTINFO directory.

```
C:\> DEL  \TESTINFO\*.* <ENTER>
```

Next, use RMDIR to remove the directory.

```
C:\> RMDIR  \TESTINFO <ENTER>
```

With the disk containing the backup files in drive A, issue the following RESTORE command:

```
C:\> RESTORE  A:  C:\TESTINFO\*.* <ENTER>
```

When RESTORE completes, the TESTINFO directory will be on your disk once again with its original files intact.

```
C:\> DIR  \TESTINFO <ENTER>
```

Clean up your disk by deleting the files in the TESTINFO directory and remove the directory using RMDIR.

If you are using DOS 6, invoke the MSBACKUP command and create the setup files INCREMEN.SET and FULLDISK.SET as previously discussed in this chapter. Next, if you have not already done so, invoke MSBACKUP and perform a full disk backup operation. Place the disks in a safe location.

Review

1. How does a complete disk backup differ from an incremental backup?
2. What is an archive bit?
3. How does the BACKUP command use the archive bit?
4. What functions do BACKUP's switches /S, /A, /M, and /L perform?
5. What is the purpose of a backup log?
6. What is a MSBACKUP setup file?
7. What is a MSBACKUP backup set?
8. What is the meaning of the MSBACKUP catalog name CD30405A.FUL?

Answers

1. A complete disk backup backs up every file on your disk. An incremental backup only backs up those files created or changed since the last BACKUP operation. You should perform a complete disk backup once a month and an incremental disk backup every day.
2. In addition to the file's name, size, and date and time stamps, DOS stores an attribute byte for each file in the file's directory entry. One of the most commonly used attributes is the archive attribute. Each time you create or change a file, DOS sets the file's archive bit, indicating the file needs to be backed up to floppy. When BACKUP successfully backs up (or archives) the file, BACKUP clears the archive bit. When you perform an incremental backup operation, BACKUP only backs up those files whose archive bit is set.
3. Each time you perform either a complete disk or incremental backup operation, BACKUP clears the archive bit for each file it successfully backs up.
4. The BACKUP command switches are as follows:

/S	Directs BACKUP to back up files contained in lower-level directories.
/A	Directs BACKUP to add backup files to an existing backup disk.
/M	Directs BACKUP to only back up those files whose archive bit is set.
/L	Directs BACKUP to log the names of each file it backs up, and the disk number containing the file.

5. Over a period of a month, the number of floppy disks containing your complete disk and incremental file backups can become quite large. If you need to restore a file from your backup floppy disks, locating the disk containing the file can consume a lot of time. If you are using DOS 3.3 or later, BACKUP's /L switch lets you create a backup log, a file containing the name of each file, and the number of its corresponding backup disk. Using the backup log, you can locate the floppy disk containing the needed file much faster.
6. An MSBACKUP setup file specifies the files that you want to back up, as well as the backup type (full, incremental, or differential) and backup options.
7. A backup set is the collection of disks and files created during a backup operation. MSBACKUP catalog files define the members of a backup set.
8. The file extension FUL indicates that the backup was a full backup operation. The filename CD30405A describes specifics about the backup. The C indicates drive C was the first drive backed up. The D indicates that drive D was the last drive backed up. The 3 specifies the backup was performed in 1993. The 0405 specifies the month and day (April 05). Finally, the A specifies that this catalog file corresponds to the first backup operation performed on that day.

Advanced Concepts: BACKUP and RESTORE

The discussion in the previous section assumed you wanted to back up each file on your disk or only those files whose archive attribute bit is set. To provide you with a little more control in choosing files to back up, the BACKUP command provides the /D and /T switches. The /D switch lets you specify a date, for BACKUP to refer to, only copying files created or changed on or after the date specified. For example, the following command backs up files created on or after July 4, 1993:

```
C:\> BACKUP  C:\*.*  A:  /S  /D:7-4-93 <ENTER>
```

In a similar way, the /T switch lets you specify a time. Users often combine the /T and /D switches like this:

```
C:\> BACKUP  C:\*.*  A:  /S  /D:7-4-93  /T:12:00 <ENTER>
```

Here, BACKUP will back up all files created after 12 noon on July 4, 1993.

As the BACKUP command provides switches to support date and time operations, so too does RESTORE. The /A and /B switches let you specify dates that control RESTORE's processing. The /A switch lets you specify a date for which RESTORE only restores files created or changed on or after the date specified. The /B switch works in just the opposite manner. Using /B, RESTORE only restores files created on or before the date specified. The following RESTORE command restores the files on drive C created on or before August 1, 1993:

```
C:\> RESTORE  A:  C:\*.*  /B:8-1-93 <ENTER>
```

Similarly, the /E and /L switches let you specify earlier and later times. By combining these switches with /A and /B, you can direct RESTORE to only restore files created on or before a specific time on a specific date. For a complete description of these date and time switches, turn to the Command Reference at the end of this book.

Although you might not use RESTORE's date and time switches on a regular basis, RESTORE provides the /P, /N, and /M switches, which you may find quite useful. RESTORE overwrites existing files on your disk, including files marked read-only, as discussed earlier in this chapter. The /P switch, however, directs RESTORE to display the following prompt before overwriting a read-only file.

```
WARNING! File FILENAME.EXT
is a read-only file
Replace the file (Y/N)?
```

/P also displays the following prompt if RESTORE detects that it is about to overwrite a file that was changed subsequent to backing up.

```
WARNING! File FILENAME.EXT
was changed after it was backed up
Replace the file (Y/N)?
```

Using the /P switch, you can reduce the possibility of RESTORE accidentally overwriting a needed file.

The /N switch provides an extra layer of protection by directing RESTORE only to copy those files that are no longer on your hard disk. In this way, RESTORE will not overwrite an existing file.

The /M option directs RESTORE to only restore those files changed since the last BACKUP operation. Assume, for example, your system crashes, leaving several files in a partially changed state. Using the /M option, you can quickly restore the disk to its state immediately following the last backup operation.

If you are using DOS 5 or later, the /D switch directs RESTORE to display a list of matching files without actually restoring the files. For example, if you are trying to remember the name of a file containing a specific letter, you might use the /D switch as follows:

```
C:\> RESTORE  A:  C:\*.LTR  /S  /D <ENTER>
```

Using BACKUP and RESTORE with Large Files

BACKUP and RESTORE use a unique storage format. One of the advantages this format provides is that BACKUP gives you a way of copying a file whose size is larger than your floppy disk. Assume, for example you need to give another user a copy of the file TWOMEG.DAT, which contains more than 2Mb of data. If you try to use the COPY command to put the file on a floppy disk, the floppy disk will run out of space:

```
C:\> COPY  TWOMEG.DAT  A: <ENTER>
Insufficient disk space
        0 File(s) copied
```

Using the BACKUP command, you can copy parts of a file to different floppy disks.

```
C:\> BACKUP  TWOMEG.DAT  A: <ENTER>
```

Later, using RESTORE, the other user can place the file on his or her hard disk.

```
C:\> RESTORE  A:  C:TWOMEG.DAT <ENTER>
```

Using BACKUP and RESTORE in this way, you can copy large files without having to buy third-party data-compression software.

Using BACKUP and RESTORE in Batch Files

To simplify the task of performing monthly and incremental backups, many users create the batch files MONTHLY.BAT and DAILY.BAT. The batch files use BACKUP's exit status values, defined in Table 22-4.

The following batch file, MONTHLY.BAT, uses BACKUP to perform a complete disk backup. The batch file assumes you are using DOS 3.3 or later and that you have a directory named BACKUP for storing the backup log. If you aren't using DOS 3.3 or later, remove the /L switch from the BACKUP command, along with "C:\BACKUP\BACKUP.LOG" (the commands that save and rename the latest version of BACKUP.LOG).

```
@ECHO OFF
BACKUP C:\*.* A: /S /L:C:\BACKUP\BACKUP.LOG

IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 ECHO Successful backup
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO No files to back up
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO File sharing conflicts
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO Ended by user CTRL-C
IF ERRORLEVEL 4 ECHO Error in processing
```

In a similar way, the following batch file, DAILY.BAT, directs BACKUP to perform an incremental backup. If you aren't using DOS 3.3 or later, remove the /L switch and the BACKUP.LOG commands ("C:\BACKUP\BACKUP.LOG").

```
@ECHO OFF
BACKUP C:\*.* A: /S /A /M /L:C:\BACKUP\BACKUP.LOG

IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 ECHO Successful backup
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO No files to back up
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO File sharing conflicts
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO Ended by user CTRL-C
IF ERRORLEVEL 4 ECHO Error in processing
```

Exit Status	Meaning
0	Successful backup operation
1	No files found to back up
2	File-sharing conflicts prevented a complete backup
3	User termination via CTRL-C
4	Fatal processing error

Table 22-4. BACKUP'S Exit Status Values

Last, there may be times you want to use the RESTORE command within your batch files. Table 22-5 describes RESTORE's exit status values.

Exit Status	Meaning
0	Successful file restoration
1	No files found to restore
3	User termination via CTRL-C
4	Fatal processing error

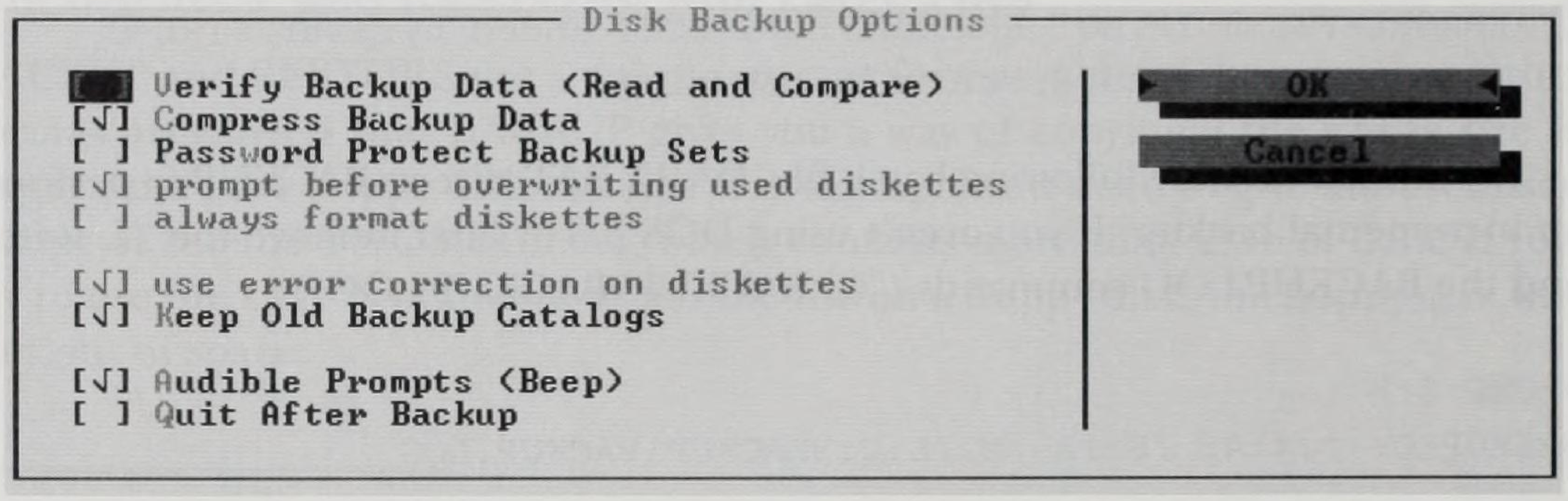
Table 22-5. *RESTORE's Exit Status Values*

Advanced Concepts: MSBACKUP

After you have used MSBACKUP for a while, you might want to take advantage of several of its advanced options. This section examines MSBACKUP's backup and restore options and MSBACKUP's compare operation.

MSBACKUP's Backup Options

To set or display MSBACKUP's backup options, select Options from the Backup screen. MSBACKUP will display the Disk Backup Options dialog box.



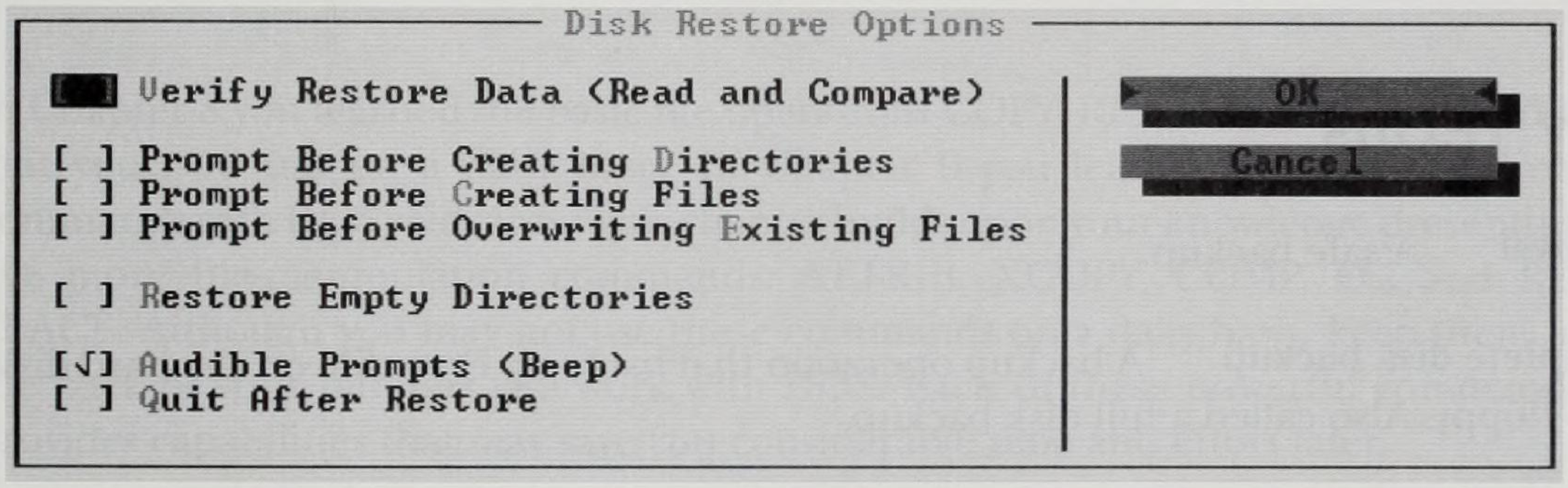
```
Disk Backup Options

[ ] Verify Backup Data (Read and Compare)
[√] Compress Backup Data
[ ] Password Protect Backup Sets
[√] prompt before overwriting used diskettes
[ ] always format diskettes

[√] use error correction on diskettes
[√] Keep Old Backup Catalogs

[√] Audible Prompts (Beep)
[ ] Quit After Backup
```

To select a specific option, highlight the option using your keyboard arrow keys and press the SPACEBAR. MSBACKUP will place a check mark within the option box indicating the option has been selected. To remove an option, perform the same steps.

If you select the Verify option, MSBACKUP will read each file it writes to the floppy disk to ensure the information was recorded correctly. Although the verification increases the reliability of your backups, most users feel that because disk record errors are rare, the verify operation is unnecessary and only consumes time. If you select the Compress option, MSBACKUP will compress the data it stores on the backup floppy. As such, you reduce the number of floppy disks required. The tradeoff is that the compression operations cause the backup operation to require more time. The error correction on disk option directs MSBACKUP to place additional information on the backup floppies, which it can later use to help rebuild the files during a restore operation should the disk become damaged. Using the error correction improves the reliability of your backups but also increases the amount of time and number of floppies the backup will require. The remainder of the backup options are fairly straightforward. If you have a specific question on an option, use MSBACKUP's online help facility.

MSBACKUP's Restore Options

To view or set MSBACKUP's restore options, select Options from the Restore screen. MSBACKUP will display the Disk Restore Options dialog box.



Disk Restore Options

[■] Verify Restore Data (Read and Compare)

[ ] Prompt Before Creating Directories
[ ] Prompt Before Creating Files
[ ] Prompt Before Overwriting Existing Files

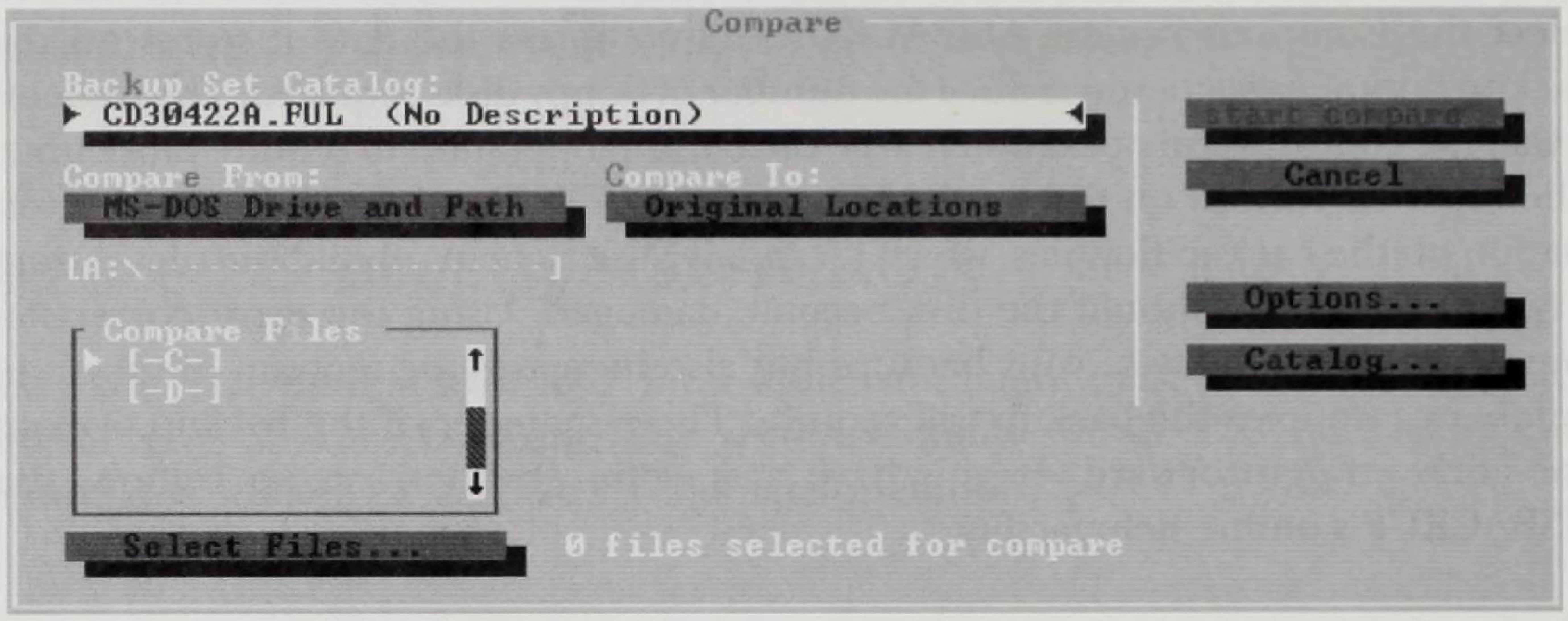
[ ] Restore Empty Directories

[√] Audible Prompts (Beep)
[ ] Quit After Restore

As before, to select or deselect an option, highlight the option using your keyboard's arrow keys and press the SPACEBAR. Most of the restore options are straightforward. The Verify option directs MSBACKUP to read files it has restored to your hard disk to ensure they were correctly recorded. Because speed is less important than accuracy for restore operations, you should consider selecting the Verify option.

MSBACKUP Compare Operations

The MSBACKUP main menu contains a Compare option, which lets you compare the contents of the files backed up to floppy with the files on your hard disk. Some users will use the Compare option to be doubly sure that their backup operation was successful. You can also use Compare to determine which files on your disk have changed since the last backup operation. When you select the Compare option, MSBACKUP will display the Compare screen, as shown in the following.



Backup Set Catalog:
CD30422A.FUL (No Description)

Compare From:
MS-DOS Drive and Path

Compare To:
Original Locations

[A:\]

0 files selected for compare

Performing a compare operation is similar to performing a restore in that you first select a catalog file. When you begin the compare operation, MSBACKUP will compare each file contained on the backup floppy to its counterpart on your hard disk.

Key Terms

Archival A safe backup.

Complete disk backup A backup operation that backs up every file on the hard disk to floppy. Also called a full disk backup.

Incremental backup A backup operation that copies only those files created or changed since the last backup.

Differential backup A backup operation that backs up every file created or changed since the last full disk backup.

Log file A list of the files on your backup disks that tells which disk each file is on.

Chapter 23

Advanced File Manipulation

In Chapter 8 you learned the basic file operations COPY, RENAME, DEL, and TYPE that you now use on a daily basis. In Chapter 9 you learned how to use these commands with DOS wildcards. This chapter builds upon your knowledge presenting five more file-manipulation commands: ATTRIB, XCOPY, COMP, FC, and REPLACE. Although you may not use these commands on a daily basis, keep them in the back of your mind as you work with DOS. Each of these powerful commands provides capabilities that may save you considerable time and effort later.

Manipulating File Attributes with ATTRIB

A directory is a list of filenames, extensions, sizes, and date and time stamps. In Chapter 22, you learned that each directory entry also contains an attribute byte that provides DOS with more information about the file. You also learned each time you create a file or change a file's contents, DOS sets the file's archive-required attribute, or archive bit, which tells DOS you have not yet backed up the file's new contents to another disk. By searching your directories for files with the archive bit set, the BACKUP command (or DOS 6 MSBACKUP) can selectively back up only those files created or changed since the last backup operation.

The ATTRIB command is an external command that lets you set or display file attributes. Using ATTRIB, you can set or clear a file's archive and read-only attributes.

As you have read, the archive attribute specifies whether or not BACKUP should back up the file to floppy. A file's read-only attribute prevents DOS from changing, deleting, or overwriting the file's contents. If you are using DOS 5 or later, ATTRIB also lets you set or clear a file's hidden and system attributes. The "Advanced Concepts" section discusses these two file attributes. Do not use ATTRIB to set or clear these attributes until you are very familiar with DOS and have thoroughly read and understand the discussion of their use.

Displaying File Attributes

ATTRIB sets and displays file attributes. To begin, display the attributes of the files in your DOS directory with the following command:

```
C:\> ATTRIB  \DOS\*.* <ENTER>
```

ATTRIB will list each filename, preceding the filename with the letters A or R if the file's archive or read-only attributes are set.

```
C:\> ATTRIB  \DOS\*.* <ENTER>
                C:\DOS\FORMAT.COM
                C:\DOS\SHARE.EXE
                C:\DOS\XCOPY.EXE
  A             C:\DOS\DOSSHELL.VID
                C:\DOS\DOSSHELL.INI
                C:\DOS\DOSSHELL.COM
  A             C:\DOS\DOSSHELL.EXE
  A             C:\DOS\DOSSHELL.GRB
  A             C:\DOS\DOSSWAP.EXE
  A             C:\DOS\EXE2BIN.EXE
```

The letter A states the file's archive bit is set. In this case, the letter R does not appear, indicating none of the files are read-only.

In DOS 5 or later, ATTRIB will also display the letter S if the file's system attribute is set and H if the file is a hidden file. In DOS 3.3 or later, the /S switch directs ATTRIB to also process directories below the directory specified. For example, the following command directs ATTRIB to display the attributes for every file on your current (drive C) disk:

```
C:\> ATTRIB  \*.*  /S <ENTER>
```

If the filenames and attributes scroll past you faster than you can read them, use the DOS pipe operator discussed in Chapter 21 to display the command's output a screenful at a time using MORE.

```
C:\> ATTRIB  \*.*  /S  |  MORE <ENTER>
```

If you have recently performed a backup operation, note that only those files you have created or changed since the backup operation have their archive bit set.

Setting a File's Attributes

In Chapter 22 you learned that each time BACKUP (or the DOS 6 MSBACKUP command) successfully backs up a file to another disk, it clears the file's archive bit. As you will learn, there are times when it is convenient to set or clear the file's archive bit yourself. To set a file's archive attribute, invoke ATTRIB with +A and the filename.

```
C:\> ATTRIB  +A FILENAME.EXT <ENTER>
```

To clear a file's archive bit, use –A. The following ATTRIB command sets the archive bit of the files in your DOS directory:

```
C:\> ATTRIB  +A  \DOS\*.* <ENTER>
```

If you invoke ATTRIB to display the file attributes, ATTRIB will precede each filename with an uppercase A.

```
C:\> ATTRIB  \DOS\*.* <ENTER>
  A         C:\DOS\FORMAT.COM
  A         C:\DOS\SHARE.EXE
  A         C:\DOS\XCOPY.EXE
  A         C:\DOS\DOSSHELL.VID
  A         C:\DOS\DOSSHELL.INI
  A         C:\DOS\DOSSHELL.COM
```

In a similar way, the ATTRIB command, shown in the following example, clears the archive bit of the files in the DOS directory.

```
C:\> ATTRIB  -A  \DOS\*.* <ENTER>
```

As you learned in Chapter 22, an incremental backup operation uses the archive bit to select the files to backup. If, for some reason, you want BACKUP to exclude one or more files, you can clear the file's archive bit manually using –A. Later in this chapter you will examine the XCOPY command, which provides extended capabilities for copying files. As you will learn, you can easily copy a directory of files to two or more floppy disks by combining ATTRIB and XCOPY.

A *read-only file* is a file whose contents DOS can read, but cannot change, delete, or overwrite. If you have files whose contents seldom change, setting the file to read-only protects it from being accidentally deleted or overwritten. All of your EXE or COM files, for example, are excellent candidates for read-only files because their contents never change.

The ATTRIB command lets you set one or more files to read-only using +R as shown here:

```
C:\> ATTRIB  +R FILENAME.EXT <ENTER>
```

To better understand how setting a file to read-only protects your files, create the file TEST.DAT.

```
C:\> COPY  CON  TEST.DAT <ENTER>

This is a read-only test file. <ENTER>
^Z <ENTER>
        1 File(s) copied
```

Using ATTRIB +R as shown here, set the file to read-only:

```
C:\>  ATTRIB  +R  TEST.DAT <ENTER>
```

DOS can still use read-only files for TYPE and COPY operations, as they only read the file.

```
C:\> TYPE  TEST.DAT <ENTER>

This is a read-only test file.
```

DOS cannot delete a read-only file. If you try, DOS displays the "Access denied" error message.

```
C:\> DEL  TEST.DAT <ENTER>
Access denied
```

Likewise, if you try to overwrite the file's contents using COPY, the command will fail as shown here:

```
C:\> COPY  FILENAME.EXT  TEST.DAT <ENTER>

Access denied - TEST.DAT
      0 File(s) copied
```

There may be times you want to change the file or delete the file when it is no longer needed. Using ATTRIB –R, you can remove the file's read-only attribute.

```
C:\> ATTRIB  -R  TEST.DAT <ENTER>
```

When the file is no longer read-only, you can delete the file.

```
C:\> DEL  TEST.DAT <ENTER>
```

The ATTRIB command lets you combine attribute settings in the same command. For example, the following command sets the files with the EXE extension in your DOS directory to read-only, while clearing the file's archive bit:

```
C:\> ATTRIB  +R  -A  \DOS\*.EXE <ENTER>
```

XCOPY's Extended Copy Capabilities

Chapter 8 introduced the COPY command. Since then, you have used COPY as one of your fundamental DOS commands. As you work with DOS for some time, you will run into situations where COPY makes an operation more difficult than it needs to be. The most common case that frustrates users is copying a directory of files to multiple floppy disks.

Assume, for example, you want to copy the files in your DOS directory to floppy disks. If you are using 360Kb floppy disks, the copy operation will require several floppies. To begin the operation, invoke COPY as shown here:

```
C:\> COPY  \DOS\*.*  A: <ENTER>
```

COPY will begin copying files from the directory to your disk in drive A, displaying the name of the file it is currently copying. Eventually, your floppy disk will run out of space and COPY will display the "Insufficient disk space" error message.

```
C:\> COPY  \DOS\*.*  A: <ENTER>
C:\DOS\EGA.SYS
C:\DOS\DISPLAY.SYS
C:\DOS\FORMAT.COM
C:\DOS\ANSI.SYS
C:\DOS\COUNTRY.SYS
C:\DOS\HIMEM.SYS
C:\DOS\KEYB.COM
C:\DOS\KEYBOARD.SYS
C:\DOS\MODE.COM
C:\DOS\SETVER.EXE
C:\DOS\SYS.COM
Insufficient disk space
       10 file(s) copied
```

In this case, COPY ran out of space when it tried to copy the file SYS.COM. To continue the file copying you insert a new disk in drive A. Here's where the trouble begins. COPY does not provide a way for you to continue the file copy operation with the last file copied. Your only option is to copy the files one by one, beginning with the first uncopied file.

If you are using DOS 3.2 or later, the XCOPY command extends the COPY command's capabilities providing you with more control over your file copy operations. For example, by combining XCOPY with the ATTRIB command, you can easily copy a directory of files to multiple floppy disks using XCOPY's /M switch.

The XCOPY /M switch directs XCOPY to behave much like the BACKUP command, copying only those files whose archive bit is set. When XCOPY successfully copies a file, XCOPY clears the archive-required bit. Using the /M switch in this way, XCOPY essentially places a bookmark at the last file copied. Should a floppy disk run out of space, you can insert a new floppy disk and repeat the XCOPY /M command. XCOPY will resume the copy operation with the first file whose archive bit is set, as desired.

To copy a directory of files using XCOPY /M, you must first set each file's archive bit. To copy the files in your DOS directory to drive A, for example, first set the file attributes using ATTRIB as shown here:

```
C:\> ATTRIB  +A  \DOS\*.* <ENTER>
```

Next, invoke XCOPY using /M:

```
C:\> XCOPY  \DOS\*.*  A:  /M <ENTER>
```

Like the COPY command, XCOPY will display the name of each file it is copying. As before, when the floppy disk fills, XCOPY will display the "Insufficient disk space" message.

```
Insufficient disk space

      10 File(s) copied
```

When this occurs, insert a new floppy disk in drive A and repeat the same XCOPY command.

```
C:\> XCOPY  \DOS\*.*  A:  /M <ENTER>
```

Because XCOPY has been clearing the archive bit of each file it successfully copies, XCOPY can pick up after the last file copied. Depending on the number of files in your directory, you may have to repeat the XCOPY command several times.

XCOPY provides the /S switch, which lets you copy files contained in directories below the directory specified. Using /S and /M, you might copy all the files in your WordPerfect directory, WP51, and directories below the WordPerfect directory with the following commands:

```
C:\> ATTRIB  +A  \WP51\*.*  /S <ENTER>
C:\> XCOPY  \WP51\*.*  A:  /S  /M <ENTER>
```

This book's Command Reference describes XCOPY in detail. Table 23-1 briefly describes XCOPY's switches.

Switch	Function
/A	Copies only files whose archive attribute is set. Does not clear the attribute.
/D: *MM-DD-YY*	Copies only files created or changed on or after the date specified.
/E	Copies empty directories to the target disk (creates the directory). Use /E in combination with /S.
/M	Copies only files whose archive attribute is set. Clears the bit after successfully copying the file.
/P	Prompts you for each file individually to confirm you want to copy the file.
/S	Copies all directories below the directory specified.
/V	Verifies that each file was copied correctly.
/W	Prompts you to press a key to begin the copy operation, letting you first change disks.

Table 23-1. *XCOPY Command-Line Switches*

Comparing Two Files with COMP

After you've worked with DOS for some time, it's not uncommon for two directories to contain files with the same name. If the files are ASCII files, you may be able to use TYPE or your word processor to determine if the files are the same. If you can't tell if the files differ, you can use the COMP command.

NOTE *DOS 6 does not support the COMP command.*

COMP is an external DOS command that compares the contents of two files. In its most general form, you invoke COMP with the two filenames you want to compare, as shown here:

```
C:\> COMP FILENAME.ONE  FILENAME.TWO <ENTER>
```

COMP compares each file's contents byte for byte. If the files are exactly the same, COMP displays the following message:

```
Files compare OK
Compare more files (Y/N) ?
```

To compare other files, type **Y** and press ENTER.

COMP will prompt you to type in the next two filenames. To return to the DOS prompt, type **N** and press ENTER.

If the files differ, COMP displays the byte number (or offset) of the first ten differences along with the corresponding byte values. Unfortunately, COMP displays these values using the *hexadecimal* or base 16 number system, as shown here:

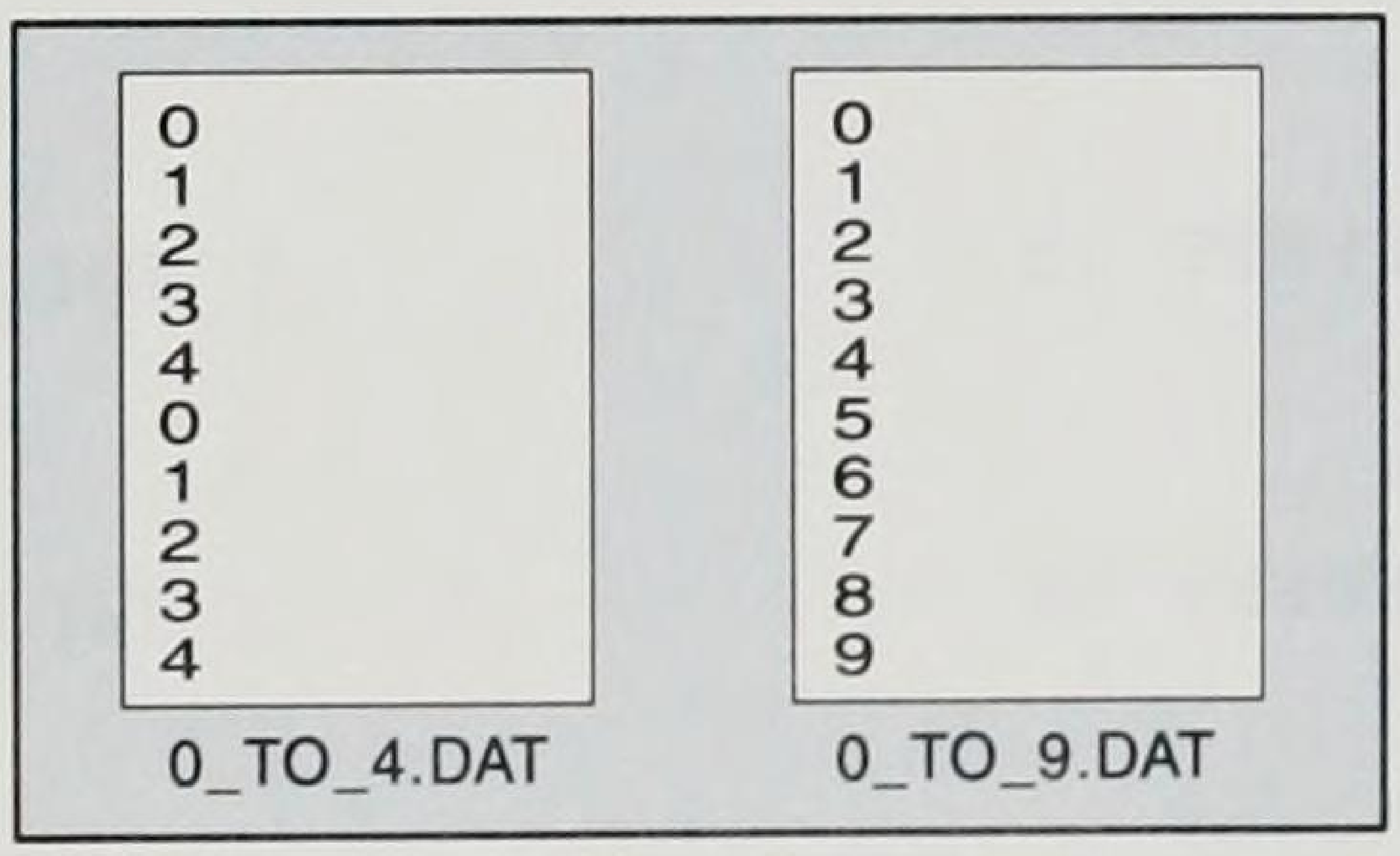
```
Compare error at OFFSET 1A
File 1 = F3
File 2 = C2
```

Computer programmers use hexadecimal because it is convenient to represent the 0 and 1 value combinations used by the computer. Most users, unfortunately, aren't conversant with hexadecimal. If you are using COMP to determine if two files are the same, you may not care about how the files differ, and you can ignore the hexadecimal values. If you want to know how the files actually differ, you can use the FC command, the next command discussed. If you are using DOS 5, COMP has become much more user friendly.

Using COMP in DOS 5

As you have just learned, COMP's shortcoming is that it displays the offset and file differences as hexadecimal values. Under DOS 5, the /D switch directs COMP to display the differences as decimal values. Similarly, the /A switch directs COMP to display the values in ASCII.

Assume, for example, the files 0_TO_4.DAT and 0_TO_9.DAT contain the following:



```
0
1
2
3
4
0
1
2
3
4
```

0_TO_4.DAT

```
0
1
2
3
4
5
6
7
8
9
```

0_TO_9.DAT

By default, COMP will display the offset and file differences in hexadecimal, when you compare the files as shown in the following example:

```
C:\> COMP  0_TO_4.DAT  0_TO_9.DAT <ENTER>
Comparing 0_TO_4.DAT and 0_TO_9.DAT...
Compare error at OFFSET F
file1 = 30
file2 = 35
Compare error at OFFSET 12
file1 = 31
file2 = 36
Compare error at OFFSET 15
file1 = 32
file2 = 37
Compare error at OFFSET 18
file1 = 33
file2 = 38
Compare error at OFFSET 1B
file1 = 34
file2 = 39

Compare more files (Y/N) ?
```

Using the DOS 5 /D switch, COMP displays the following:

```
C:\> COMP  0_TO_4.DAT  0_TO_9.DAT  /D <ENTER>
Comparing 0_TO_4.DAT and 0_TO_9.DAT...
Compare error at OFFSET F
file1 = 48
file2 = 53
Compare error at OFFSET 12
file1 = 49
file2 = 54
Compare error at OFFSET 15
file1 = 50
file2 = 55
Compare error at OFFSET 18
file1 = 51
file2 = 56
Compare error at OFFSET 1B
file1 = 52
file2 = 57

Compare more files (Y/N) ?
```

If you examine the chart in Appendix B, you'll find the decimal representations for the ASCII symbols 0 through 9 are the values 48 through 57.

The /A switch, in this case, provides the most understandable output.

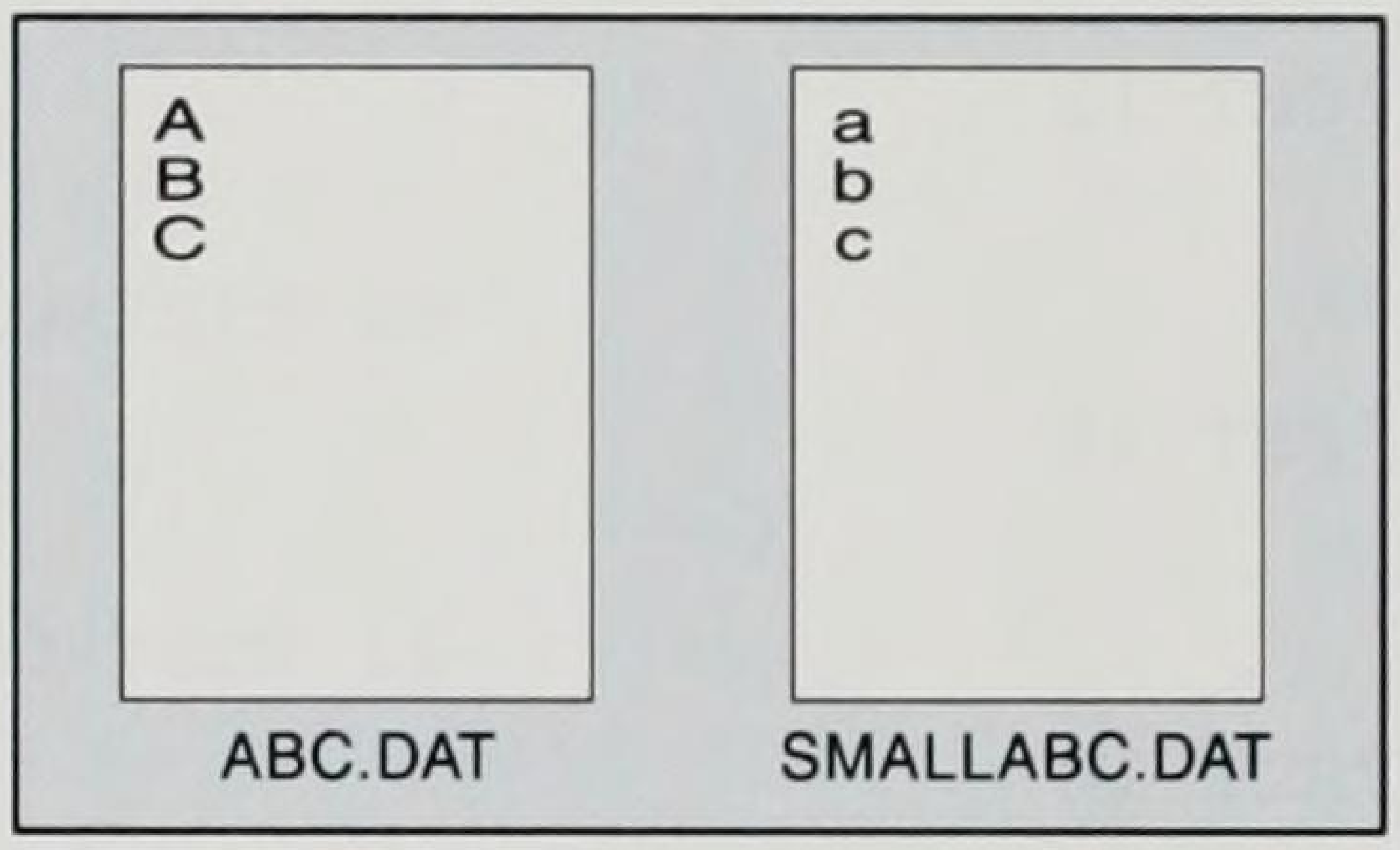
```
C:\> COMP  0_TO_4.DAT  0_TO_9.DAT  /A <ENTER>
Comparing 0_TO_4.DAT and 0_TO_9.DAT...
Compare error at OFFSET F
file1 = 0
file2 = 5
Compare error at OFFSET 12
file1 = 1
file2 = 6
Compare error at OFFSET 15
file1 = 2
file2 = 7
Compare error at OFFSET 18
file1 = 3
file2 = 8
Compare error at OFFSET 1B
file1 = 4
file2 = 9

Compare more files (Y/N) ?
```

Note that even though you use the /D or /A switch, COMP still displays the offset in hexadecimal. Using the /L switch, you can direct COMP to display the line number of each difference, as follows:

```
C:\> COMP  0_TO_4.DAT  0_TO_9.DAT  /L <ENTER>
Comparing 0_TO_4.DAT and 0_TO_9.DAT...
Compare error at LINE 6
file1 = 30
file2 = 35
Compare error at LINE 7
file1 = 31
file2 = 36
Compare error at LINE 8
file1 = 32
file2 = 37
Compare error at LINE 9
file1 = 33
file2 = 38
```

```
Compare error at LINE 10
file1 = 34
file2 = 39

Compare more files (Y/N) ?
```

COMP does not consider upper- and lowercase letters as the same. Assume, for example, the files ABC.DAT and SMALLABC.DAT contain the following:



COMP considers the files' contents different.

```
C:\> COMP  ABC.DAT  SMALLABC.DAT <ENTER>
Comparing ABC.DAT and SMALLABC.DAT...
Compare error at OFFSET 0
file1 = 41
file2 = 61
Compare error at OFFSET 3
file1 = 42
file2 = 62
Compare error at OFFSET 6
file1 = 43
file2 = 63

Compare more files (Y/N) ?
```

Using the /C switch, however, you can direct COMP to consider upper- and lowercase letters as the same:

```
C:\> COMP  ABC.DAT  SMALLABC.DAT  /C <ENTER>
Comparing ABC.DAT and SMALLABC.DAT...
Files compare OK

Compare more files (Y/N) ?
```

Last, by default, COMP will not compare files whose sizes differ. If you try to compare files of different sizes, COMP displays the following error message:

```
C:\> COMP  SMALLABC.DAT  0_TO_4.DAT <ENTER>
Comparing SMALLABC.DAT and 0_TO_4.DAT...
Files are different sizes

Compare more files (Y/N) ?
```

Using the /N: *n* switch, you can direct COMP to compare the first *n* lines of two files, even though their sizes may differ.

As you can see, COMP has come a long way in DOS 5. If you are not now using DOS 5, you'll probably want to compare files using the FC command.

Comparing Files with FC

If you are using MS-DOS 2 or later, the FC command provides a more flexible file-comparison utility than COMP. FC is not available in IBM PC-DOS. You learned that COMP compares files on a byte-by-byte basis, displaying only the first ten differences between the files.

Depending on the information in the files you are comparing, COMP could encounter ten differences in the first line and stop. FC, on the other hand, is much more flexible.

The general format of the FC command is

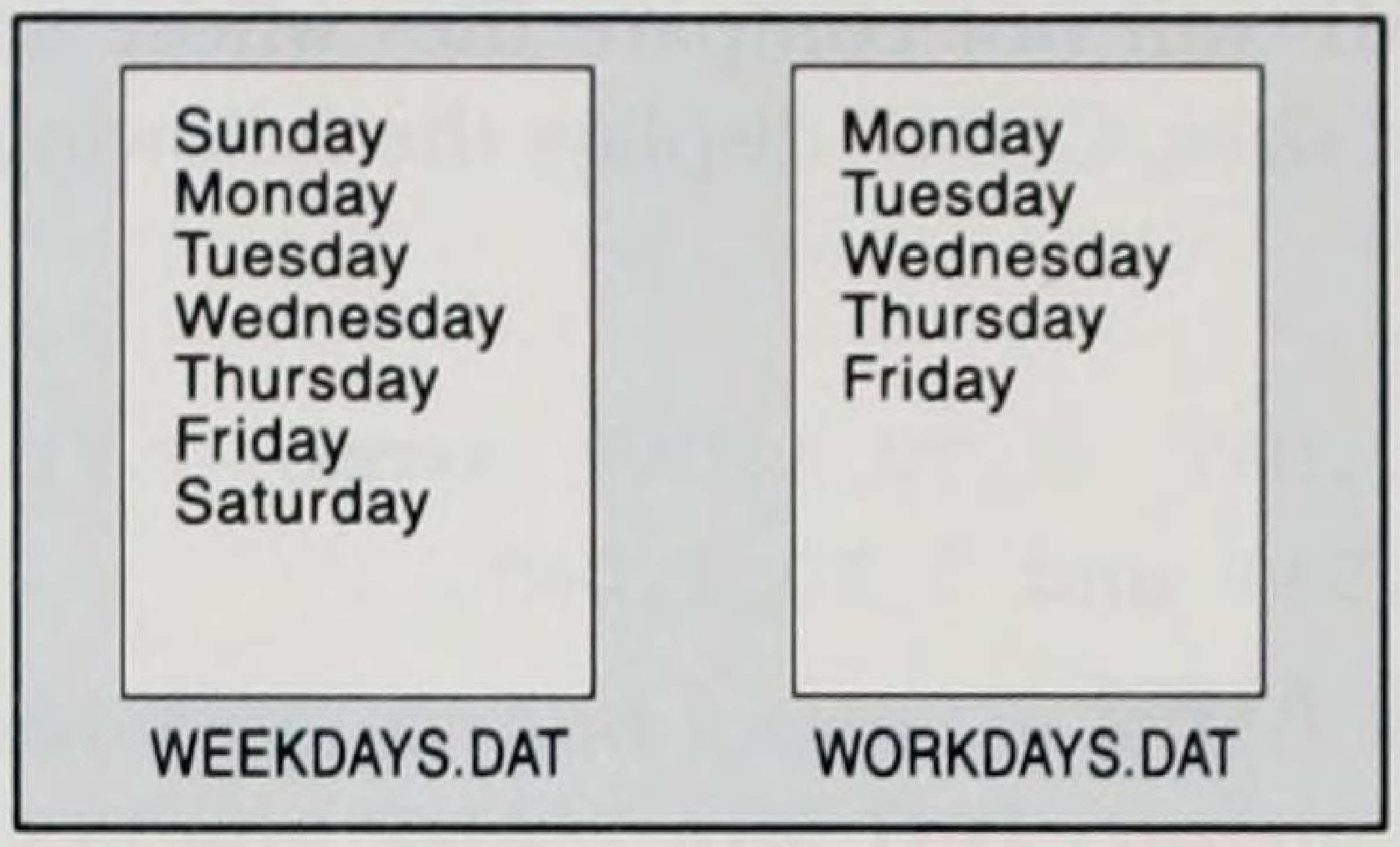
```
FC FILENAME.ONE  FILENAME.TWO [switches]
```

Use the following DIR command to determine if your DOS version has external command FC.EXE.

```
C:\> DIR  \DOS\FC.EXE <ENTER>
```

If DIR displays the "File not found" message, your DOS version doesn't support the FC command, and you should continue your reading with the REPLACE command discussed next.

To better understand how the FC command works, create the files WORKDAYS.DAT and WEEKDAYS.DAT that contain what is shown the following illustration.


Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
WEEKDAYS.DAT
Monday
Tuesday
Wednesday
Thursday
Friday
WORKDAYS.DAT

Next, compare the files using FC:

```
C:\> FC  WEEKDAYS.DAT  WORKDAYS.DAT <ENTER>
```

As you can see, FC displays the lines in each file that differ, as opposed to the byte values and offsets displayed by COMP. In this case, you can quickly determine that the two files begin and end differently. Using the /N switch, you can display the actual line numbers that differ, as shown here:

```
C:\> FC  WEEKDAYS.DAT  WORKDAYS.DAT <ENTER>
Comparing files WEEKDAYS.DAT and WORKDAYS.DAT
***** WEEKDAYS.DAT
    1:  Sunday
    2:  Monday
    3:  Tuesday
***** WORKDAYS.DAT
    1:  Monday
    2:  Tuesday
*****

***** WEEKDAYS.DAT
    7:  Saturday
***** WORKDAYS.DAT
*****
```

The FC command is very intelligent. Although the files begin with different days, FC recognizes that both files fall into the same pattern of days. Therefore, FC does not consider the file's contents as different, even though the days appear on different lines within the two files.

The Command Reference in this book examines the FC command and each of its command-line switches in detail. If you are comparing ASCII files, FC is a very powerful utility. If you are comparing non-ASCII files to determine if the files are the same, you can use either FC or COMP. Table 23-2 briefly describes FC's command-line switches.

Adding and Replacing Selected Files with REPLACE

One of the difficulties you will experience when you begin using several different applications is keeping up with new releases of upgrades to your programs. If you are using DOS 3.2 or later, the REPLACE command may simplify your task. REPLACE is an external DOS command that lets you selectively add or replace files, typically from a floppy disk to your hard disk. As you will see, REPLACE provides you with greater

Switch	Function
/A	Directs FC to display only the first and last lines in a set of lines that differ, rather than the whole set
/B	Directs FC to perform a byte-by-byte comparison similar to COMP
/C	Directs FC to treat upper- and lowercase letters as the same
/L	Directs FC to perform ASCII line-by-line comparisons. FC uses this mode by default except for files with the EXE, COM, OBJ, LIB, SYS, or BIN extension, for which FC uses /B
/LB *n*	Sets the number of consecutive lines that can differ before FC cancels the command. The default is 100
/N	Directs FC to display line numbers for lines that differ in ASCII file comparisons
/*nnn*	Sets the number of consecutive lines that must match before FC considers the two files' contents in sync. Default is 2
/T	Directs FC to not expand tabs into spaces
/W	Directs FC to compress white space. If a file contains several blanks or tabs in succession, FC is to treat all the characters as one space or tab

Table 23-2. FC Command-Line Switches

file control than simply copying the files. Assume, for example, you receive an upgrade to one of your programs that contains the following files:

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 2C28-12E3
 Directory of A:\

MENU     COM     45056 01-01-93  12:08a
DATABASE EXE       128 04-10-93   4:53p
DATABASE DAT      2816 03-03-93  10:33p
        3 file(s)      48000 bytes
                      313344 bytes free
```

Assuming the program you are upgrading is in the directory SOMEPROG, you can invoke REPLACE to update the program using the files on drive A as follows:

```
C:\> REPLACE  A:*.*  \SOMEPROG <ENTER>
```

In this case, REPLACE works much like the COPY command, simply copying the three files to the directory, as shown here:

```
C:\> REPLACE  A:*.* \SOMEPROG <ENTER>

Replacing C:\SOMEPROG\MENU.COM

Replacing C:\SOMEPROG\DATABASE.EXE

Replacing C:\SOMEPROG\DATABASE.DAT

     3 file(s) replaced
```

Even though duplicate files waste disk space, there are times when users end up with an external command file in two or more directories. Using the /S switch, you can direct REPLACE to search every directory on your disk for files matching those on the replacement disk. Using /S in this way, REPLACE automatically replaces every copy of the original files on your disk.

```
C:\> REPLACE  A:*.*  C:\  /S <ENTER>
```

Although automatically replacing every file on your disk in this way may seem convenient, it provides the chance of accidentally replacing the wrong file. To reduce the possibility of replacing the wrong file, invoke REPLACE using the /P switch. When you include /P, REPLACE will prompt you to confirm individually that you want to replace each matching file.

```
Replace FILENAME.EXT ? (Y/N)
```

In some cases, you may want REPLACE to only copy from the replacement disk those files that aren't on your hard disk.

The following command, for example, uses the /A switch to add to the SOMEPROG directory only those files on the replacement disks that aren't in the directory:

```
C:\> REPLACE  A:*.*  \SOMEPROG  /A <ENTER>
```

Depending on the program you are upgrading, you may find REPLACE to be convenient. In some cases, the upgrade instructions may include the REPLACE command you should issue. The Command Reference examines REPLACE and its switches in detail. Table 23-3 briefly describes the REPLACE command-line switches.

Switch	Function
/A	Directs REPLACE to add only those files not found on the target disk. You can't use /A with /S or /U
/P	Directs REPLACE to prompt you to confirm each file replacement or addition individually
/R	Directs REPLACE to replace files marked read-only. Without /R, the REPLACE command stops at the first read-only file
/S	Directs REPLACE to search directories below the target directory for matching files
/U	Directs REPLACE to replace only those files in the target directory that are older than the update files. Cannot be used with the /A switch
/W	Directs REPLACE to wait for you to press a key before starting the replacement, giving you time to replace disks if necessary

Table 23-3. *REPLACE Command-Line Switches*

SUMMARY

Displaying File Attributes

Every file on your disk has an attribute byte that contains the file's archive, read-only, system, and hidden attributes. To display the file's current attribute settings, invoke the ATTRIB command with the filename, as shown in the following:

```
C:\> ATTRIB FILENAME.EXT <ENTER>
```

For versions prior to DOS 5, ATTRIB will display the letter A if the file's archive bit is set and R if the file is read-only. Beginning with DOS 5, ATTRIB also displays the letter S for DOS system files and H if the file is hidden from the directory listing.

Setting File Attributes

The ATTRIB command lets you set or clear a file's archive and read-only attributes. If you are using DOS 5 or later, ATTRIB also lets you set the file's hidden or system attributes. To set a file's attributes, invoke the ATTRIB command as shown here:

```
C:\> ATTRIB  [Attribute] FILENAME.EXT <ENTER>
```

where the following attributes apply:

+A	Sets a file's archive bit
–A	Clears a file's archive bit
+R	Sets a file to read-only
–R	Sets a file to read or write access

The XCOPY Command

The XCOPY command extends the capabilities of the COPY command letting you copy files whose archive attribute is set, files created or changed after a specific date, and files that reside in directories beneath the directory specified. The format of the XCOPY command is

```
C:\> XCOPY SOURCE.EXT  TARGET.EXT  [switches] <ENTER>
```

SUMMARY

The COMP Command

The COMP command compares two files byte by byte, displaying the location and corresponding byte values of the first ten differences. The format of the COMP command is as follows:

```
C:\> COMP FILENAME.ONE  FILENAME.TWO <ENTER>
```

If you are using DOS 5, COMP provides several command-line switches that greatly improve its capabilities. DOS 6 does not support the COMP command.

The FC Command

If you are using MS-DOS 2 or later, the FC command provides a flexible file-comparison utility. Unlike COMP, which compares two files on a byte-by-byte basis, FC compares files line by line. The basic format of the FC command is

```
C:\> FC FILENAME.ONE  FILENAME.TWO  [switches] <ENTER>
```

The REPLACE Command

The REPLACE command helps you upgrade or replace files on your disk. Unlike COPY, which gives you less control over the files it copies, REPLACE lets you selectively add and replace files. REPLACE even lets you search your entire disk for files matching the upgrades. The format of the REPLACE command is

```
C:\> REPLACE SOURCE  TARGET  [switches] <ENTER>
```

Hands On

Using the following ATTRIB command, display the attributes of the files in the current directory.

```
C:\> ATTRIB  *.* <ENTER>
```

Depending on your DOS version, DOS may precede the filenames with the letters A, R, H, and S, based on the attributes set.

Using the pipe operator and MORE, display the files a screenful at a time:

```
C:\> ATTRIB  *.*  |  MORE <ENTER>
```

If you are using DOS 3.3 or later, use the /S switch to display the attributes for every file on your disk:

```
C:\> ATTRIB  \*.*  /S  |  MORE <ENTER>
```

Using the DOS output redirection operator, you can get a printed listing of the name of every file on your disk:

```
C:\>  ATTRIB  \*.*  /S  >  PRN <ENTER>
```

It is good practice to set your files whose contents don't change to read-only. Use the following commands to set the EXE and COM files in your DOS directory to read-only:

```
C:\> ATTRIB  +R  \DOS\*.EXE <ENTER>
C:\> ATTRIB  +R  \DOS\*.COM <ENTER>
```

Now create the file DELETE.ME that contains the following:

```
C:> COPY  CON  DELETE.ME <ENTER>
You can't delete read-only files <ENTER>
^Z <ENTER>
     1 File(s) copied
```

Next, use ATTRIB to set the file to read-only.

```
C:\> ATTRIB  +R  DELETE.ME <ENTER>
```

Using DEL, try to delete the read-only file.

```
C:\> DEL  DELETE.ME <ENTER>
```

When DEL fails, remove the file's read-only attribute and delete the file.

```
C:\> ATTRIB  -R  DELETE.ME <ENTER>
C:\> DEL  DELETE.ME <ENTER>
```

The most common use of the XCOPY command is to copy a directory of files to two or more floppy disks. In this case, you will use XCOPY to copy the files in your DOS directory to floppy disks in drive A.

To begin, set each file's archive bit.

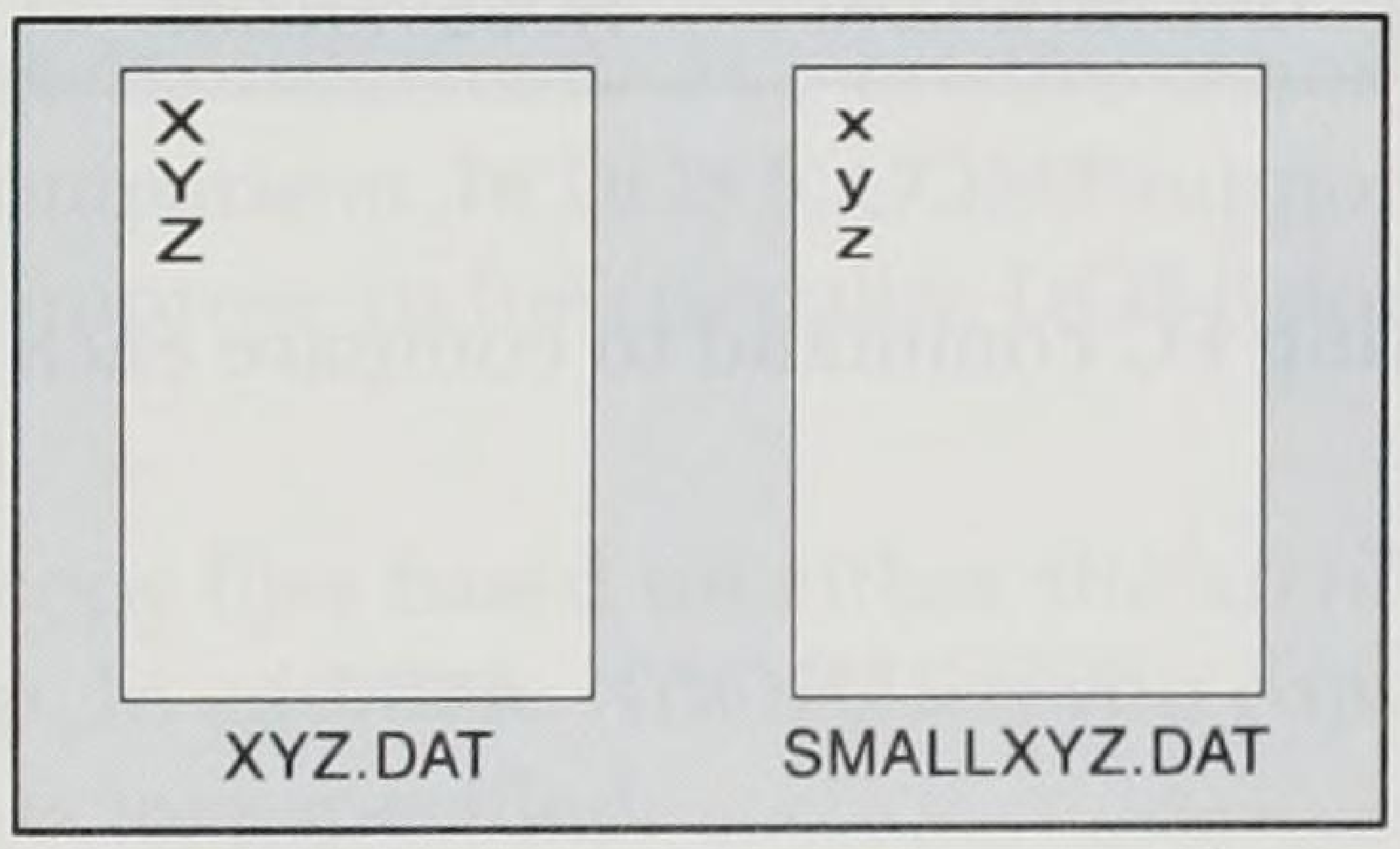
```
C:\> ATTRIB  +A  \DOS\*.* <ENTER>
```

Next, with several unused formatted floppy disks in hand, issue the following XCOPY command:

```
C:\> XCOPY  \DOS\*.*  A:  /M <ENTER>
```

As you will recall, the /M switch directs XCOPY to copy only those files whose archive bit is set. When XCOPY successfully copies a file, /M directs XCOPY to clear the file's archive bit. When the floppy disk fills, insert a second floppy disk in drive A and repeat the XCOPY command. Should you need to format additional floppy disks, do so. Because XCOPY uses the archive attribute as its bookmark, you can immediately issue XCOPY, or you can even issue the command later, at a more convenient time. Because DOS stores the file attributes in the directory on disk, you can turn off your computer or issue other DOS commands, such as FORMAT, without affecting the file attributes.

If you are using DOS 5 or earlier, create the files XYZ.DAT and SMALLXYZ.DAT that contain the following:



Next, issue the COMP command to compare each file's contents.

```
C:\> COMP  XYZ.DAT  SMALLXYZ.DAT <ENTER>
```

Using Appendix B, look up the hexadecimal values displayed and you'll find the values correspond to the letters xyz and XYZ.

If you are using DOS 5, invoke COMP with the /A switch to display the differences as ASCII characters.

```
C:\> COMP  XYZ.DAT  SMALLXYZ.DAT  /A <ENTER>
```

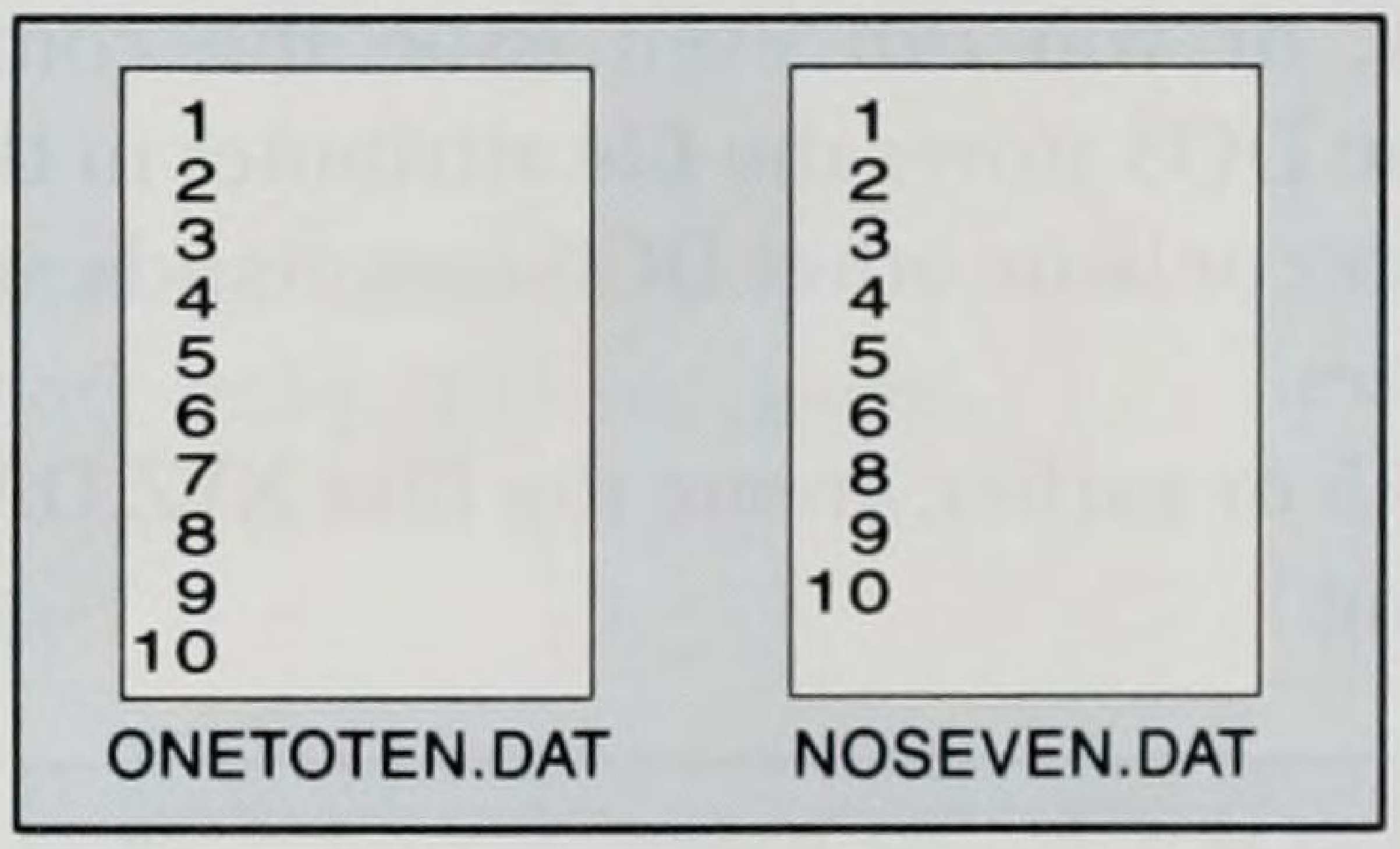
Repeat the command using the /C switch directing COMP to consider upper- and lowercase letters the same.

```
C:\> COMP  XYZ.DAT  SMALLXYZ.DAT  /C <ENTER>
```

If you are using MS-DOS 2 or later, issue the following command to determine if your DOS version supports FC.EXE:

```
C:\> DIR  \DOS\FC.EXE <ENTER>
```

If the file is present, create the files ONETOTEN.DAT and NOSEVEN.DAT, which contain the following:



```
1
2
3
4
5
6
7
8
9
10
```

ONETOTEN.DAT

```
1
2
3
4
5
6
8
9
10
```

NOSEVEN.DAT

Next, issue the following FC command to compare each file's contents:

```
C:\> FC  ONETOTEN.DAT  NOSEVEN.DAT <ENTER>
```

Repeat the command and include the /L switch directing FC to display the line numbers that differ.

```
C:\> FC  /L  ONETOTEN.DAT  NOSEVEN.DAT <ENTER>
```

Delete the files with the DAT extension you created during this "Hands On" session.

Review

1. Where does DOS store file attributes?
2. What is a read-only file?
3. What files should you set to read-only?
4. How does FC differ from COMP?
5. List capabilities XCOPY provides that COPY does not.

Answers

1. DOS stores the file attribute byte in the file's directory entry.
2. A read-only file is a file whose contents DOS can read, as the source file in a TYPE or COPY command, for example, but whose contents DOS cannot change, delete, or overwrite.
3. You should set files whose contents very rarely change, such as EXE or COM files, to read-only. By setting these files to read-only, you reduce the chance of the files being accidentally deleted or overwritten.
4. First, not all DOS versions provide FC. In fact, IBM PC-DOS does not provide FC at all. For most DOS versions, COMP compares two files displaying the first ten differences in hexadecimal. FC, on the other hand, provides a more meaningful file comparison. In DOS 5, COMP supports several new command-line switches that improve its functionality. DOS 6 does not provide the COMP command.
5. XCOPY lets you copy files based on either the archive bit being set or based on the date stamp. In addition, XCOPY lets you copy files residing in directories below the directory specified.

Advanced Concepts

Hidden files are files that exist on disk but don't appear in the directory listing. As discussed in Chapter 7, every time you create a bootable disk using FORMAT, DOS places two hidden files on the startup disk. DOS hides these files to prevent users from deleting or renaming them. If you are using DOS 5, the ATTRIB command lets you display hidden files and set files as hidden. Using ATTRIB, you can display hidden files in the root directory, as shown here:

```
C:\> ATTRIB  \*.* <ENTER>
     SH      C:\IO.SYS
     SH      C:\MSDOS.SYS
  A          C:\COMMAND.COM
             C:\AUTOEXEC.BAT
             C:\CONFIG.SYS
```

If you work in an office where your computer is accessible to other users, you may want to hide a few of your files to protect them from other users. Be careful, however, to remember you have hidden the files. Should you later have difficulty removing a directory using RMDIR, its very likely the directory may still contain hidden files.

```
C:\> RMDIR  \DIRNAME <ENTER>
Invalid path, not directory,
or directory not empty
```

Should this error message occur, invoke ATTRIB as follows:

```
C:\> ATTRIB  \DIRNAME\*.* <ENTER>
```

If hidden files exist in the directory, ATTRIB will display them. Beginning with DOS 5, the ATTRIB command also lets you set or clear a file's system attribute. DOS assigns this attribute to its two hidden root directory files. In general, you should not use the system attribute.

Leave these attribute settings for use by DOS. You can achieve the same level of file protection and privacy using the hidden attribute.

Expanding DOS 5 and 6 Files

To reduce the number of installation disks, the DOS 5 and 6 update and installation disks store files in a compressed format. If you examine these disks using DIR, you will find several files with an underscore as the last letter of the extension, such as COMMAND.CO_ or FORMAT.CO_. These files are in compressed format. When you perform the DOS upgrade or installation, the SETUP program automatically decompresses these files for you. Should you later accidentally delete a DOS file and you don't have the files backed up, you'll need to expand a copy of the compressed file yourself from the upgrade of the installation disk. The DOS 5 EXPAND command is an external command that lets you decompress a DOS 5 file. Assume, for example, you delete the file UNDELETE.EXE. Using the DIR command to examine your upgrade of installation disks, you can find the disk containing the compressed file UNDELETE.EX_. Using EXPAND, you can decompress the file into your DOS directory, as shown here:

```
C:\> EXPAND  A:UNDELETE.EX_  \DOS\UNDELETE.EXE <ENTER>
```

EXPAND supports DOS wildcards. If you need to locate several files, print a copy of the file PACKING.LST, which describes the contents of each floppy disk. PACKING.LST is not a compressed file. It resides on your upgrade disk number 1.

Key Terms

Hidden file A file that exists on your disk but does not appear in the directory listing. You cannot accidentally delete or overwrite hidden files.

System file A file whose system attribute is set, typically the files IO.SYS and MSDOS.SYS.

Read-only file A file whose contents DOS can copy or display, but cannot delete, change, or overwrite.

Compressed file A file stored in a special format to reduce the amount of disk space the file consumes. You must decompress a compressed file before you can use it. The DOS 5 EXPAND command decompresses the compressed files stored on the DOS 5 upgrade disks. If you are using DOS 6, the DECOMP command decompresses a compressed file.

Chapter 24

Customizing Your System with CONFIG.SYS

In Chapter 18 you learned that each time your system starts, DOS searches the root directory for two files named CONFIG.SYS and AUTOEXEC.BAT. As you know, AUTOEXEC.BAT is a special batch file whose commands DOS automatically executes each time your system starts. The file CONFIG.SYS, like AUTOEXEC.BAT, is a file containing single-line entries you place into the file yourself. Unlike the entries in AUTOEXEC.BAT, the CONFIG.SYS entries are not DOS commands. Instead, each CONFIG.SYS entry defines a customization setting that helps DOS configure itself in memory for your specific requirements. In fact, the name CONFIG is an abbreviation for configuration. The SYS extension indicates that the file is a DOS system file.

Understanding the CONFIG.SYS Entries

Depending on who installed DOS on your system, your root directory very likely already contains a CONFIG.SYS file. Use the following DIR command to determine if CONFIG.SYS exists:

```
C:\> DIR  \CONFIG.SYS <ENTER>
```

If the file exists, use TYPE to display its contents:

```
C:\> TYPE  \CONFIG.SYS  <ENTER>
```

In most cases, your CONFIG.SYS file will contain BUFFERS, FILES, and possibly one or more DEVICE entries, as shown here:

```
BUFFERS=30
FILES=20
DEVICE=C:\DOS\ANSI.SYS
```

Every CONFIG.SYS entry is only one line in length. Table 24-1 briefly describes the CONFIG.SYS entries.

This chapter divides the CONFIG.SYS entries into two groups: those every user may need and those best suited for advanced DOS users.

To add or change a CONFIG.SYS entry, you can use the DOS editor or your word processor. The only time DOS uses the CONFIG.SYS file is during system startup. Therefore, if you change or add a CONFIG.SYS entry, you must restart your system for DOS to use the new entry's value.

NOTE

Before changing the contents of your CONFIG.SYS file, always save a copy of the file's existing contents (see the "Hands On" section later in this chapter).

You do not need to use every entry in your CONFIG.SYS file. If you don't include one or more of the entries specified, DOS will use its own default values.

Using Disk Buffers to Improve Performance

Each time a program such as your word processor or spreadsheet reads information from a file, DOS reads one or more of the disk sectors holding the specified information from disk into an area of your computer's electronic memory called a *disk buffer.* DOS then transfers information from the disk buffer to the application's memory area as that program needs the information, as shown in .

You may be wondering why DOS first places the information into the disk buffer. Here's why. For efficiency and performance, the smallest amount of information DOS can read from or write to disk is one sector (typically 512 bytes). As an example,

Entry	Function
BREAK	Enables or disables extended CTRL-BREAK checking at system startup
BUFFERS	Defines the number of disk buffers DOS can use to improve your disk performance
COUNTRY	Defines a country-specific symbol set for international users
DEVICE	Installs software support (a device driver) for a specific hardware device
DEVICEHIGH	Similar to DEVICE, but places the device driver into reserved memory to make more conventional memory available for applications (DOS 5 or later)
DOS	Directs DOS to use reserved and high memory to make more conventional memory available for use by applications (DOS 5 or later)
DRIVPARM	Changes the parameters for a physical disk drive
FCBS	Provides support for file control blocks (FCBs) used by old DOS programs running with SHARE installed
FILES	Defines the number of files DOS can open at one time
INCLUDE	Includes the entries contained in the specified configuration block (DOS 6)
INSTALL	Loads one of four memory-resident DOS commands: KEYB, FASTOPEN, NLSFUNC, and SHARE
LASTDRIVE	Defines the last drive letter from E to Z DOS can use for a logical disk drive
MENUCOLOR	Specifies the foreground and background color of the startup menu (DOS 6)
MENUDEFAULT	Specifies the startup menu's default option (DOS 6)
MENUITEM	Specifies a system configuration menu option (DOS 6)
REM	Lets you place comments within your CONFIG.SYS file (DOS 4 or later)
SET	Assigns an entry to the DOS environment (DOS 6)
SHELL	Defines the location of the DOS command processor, typically COMMAND.COM
STACKS	Provides additional RAM stack space to handle hardware interrupts
SUBMENU	Specifies a lower-level menu of startup options
SWITCHES	Forces an enhanced keyboard to generate scancode values corresponding to a conventional keyboard (DOS 4 or later)

Table 24-1. CONFIG.SYS Entries

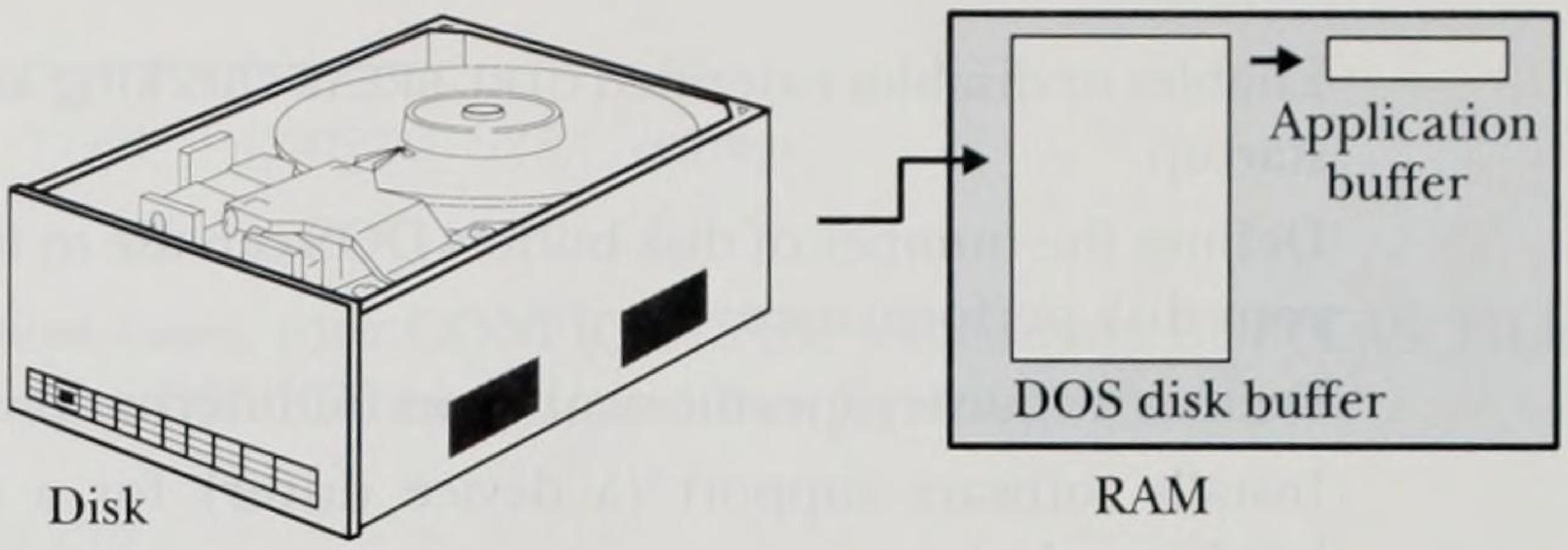


Figure 24-1. *Information flows from disk to memory buffers, then to application's memory*

assume you are running a database program that stores a simple mailing list. Each entry in this mailing list requires 64 bytes, as shown in Figure 24-2.

When you want to print a mailing label for the first database entry, the database program reads the entry from disk. When the database program asks DOS to read the entry, DOS must read and transfer a 512-byte sector from disk to memory. Because the size of each database entry is only 64 bytes, DOS actually transfers the first eight database entries from disk to memory (8 × 64 = 512 bytes), as shown in Figure 24-3.

Here's how disk buffers improve your system performance. Assume that after printing the mailing label for the first entry, you decide to print a label for the second. When the database program asks DOS to read the second entry from disk, DOS doesn't have to read the information from disk, because the entry is already in the disk buffer.

Although hard disks are becoming faster, they are still much slower than your computer's electronic memory. A general rule to follow is that reducing the number

Name **20 bytes**	**Address** **22 bytes**	**City** **15 bytes**	**State** **2 bytes**	**ZIP code** **5 bytes**
Beth Arnold	2013 South 1st	Phoenix	AZ	85022
Ken Davis	Route 2 Box 3	Kent	WA	98032
Bill Jones	1531 South 9th	Medford	MA	02155
Alex Klause	1433 Rainbow	S. Spring	MD	20910

Figure 24-2. *A simple database with 64-byte entries*

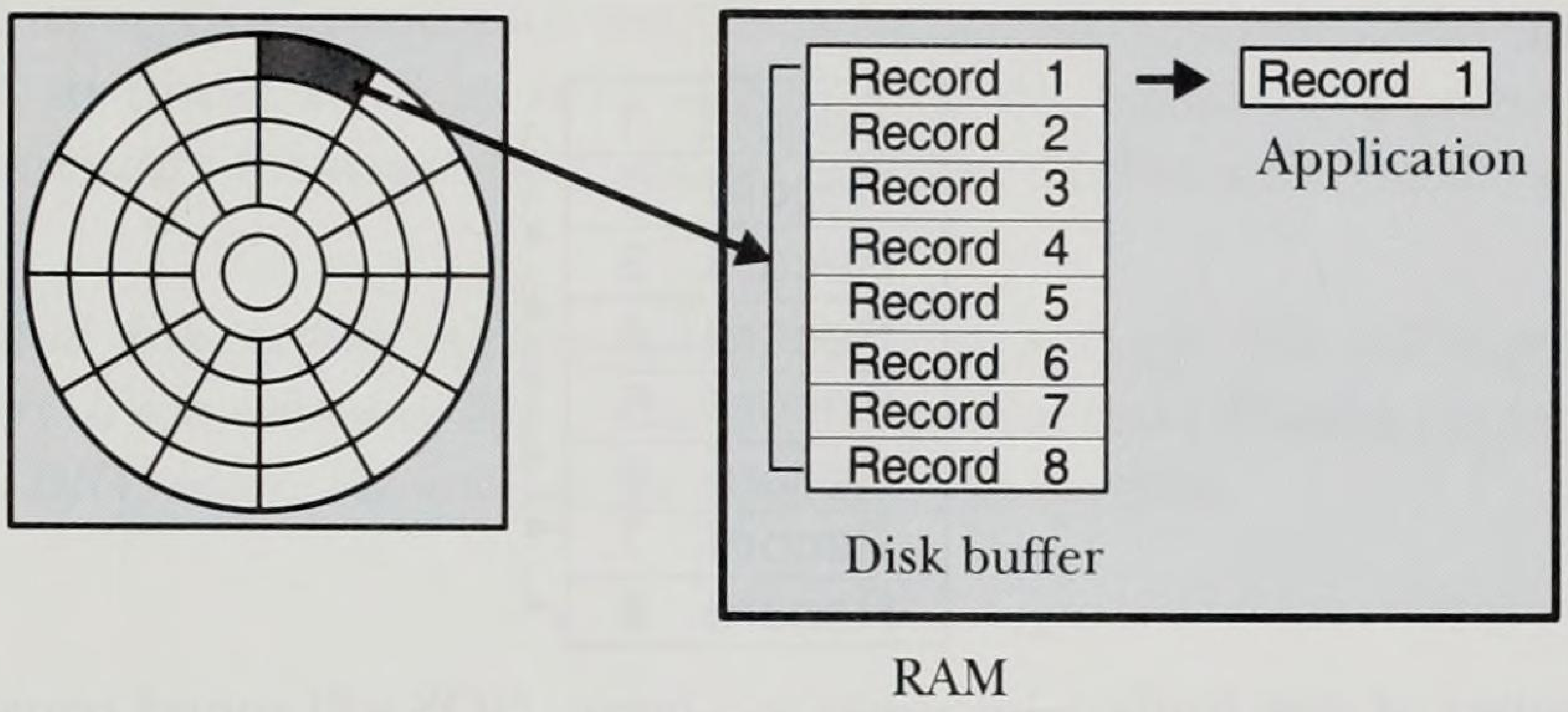


Figure 24-3. *The smallest unit of information DOS can transfer to or from disk is a sector*

of disk input or output operations will improve a program's speed. Using the database example, if you decide to print mailing labels for the first eight entries, DOS only has to perform one slow disk read. Because the information was already in the disk buffer, DOS could very quickly provide the database program with entries two through eight.

The CONFIG.SYS BUFFERS entry lets you specify the number of disk buffers you want DOS to use. To a point, the more buffers you provide, the better your system performance. Depending on your system configuration, the number of disk buffers you provide will differ slightly. Table 24-2 provides recommended buffer settings.

The following CONFIG.SYS entry directs DOS to use 30 disk buffers:

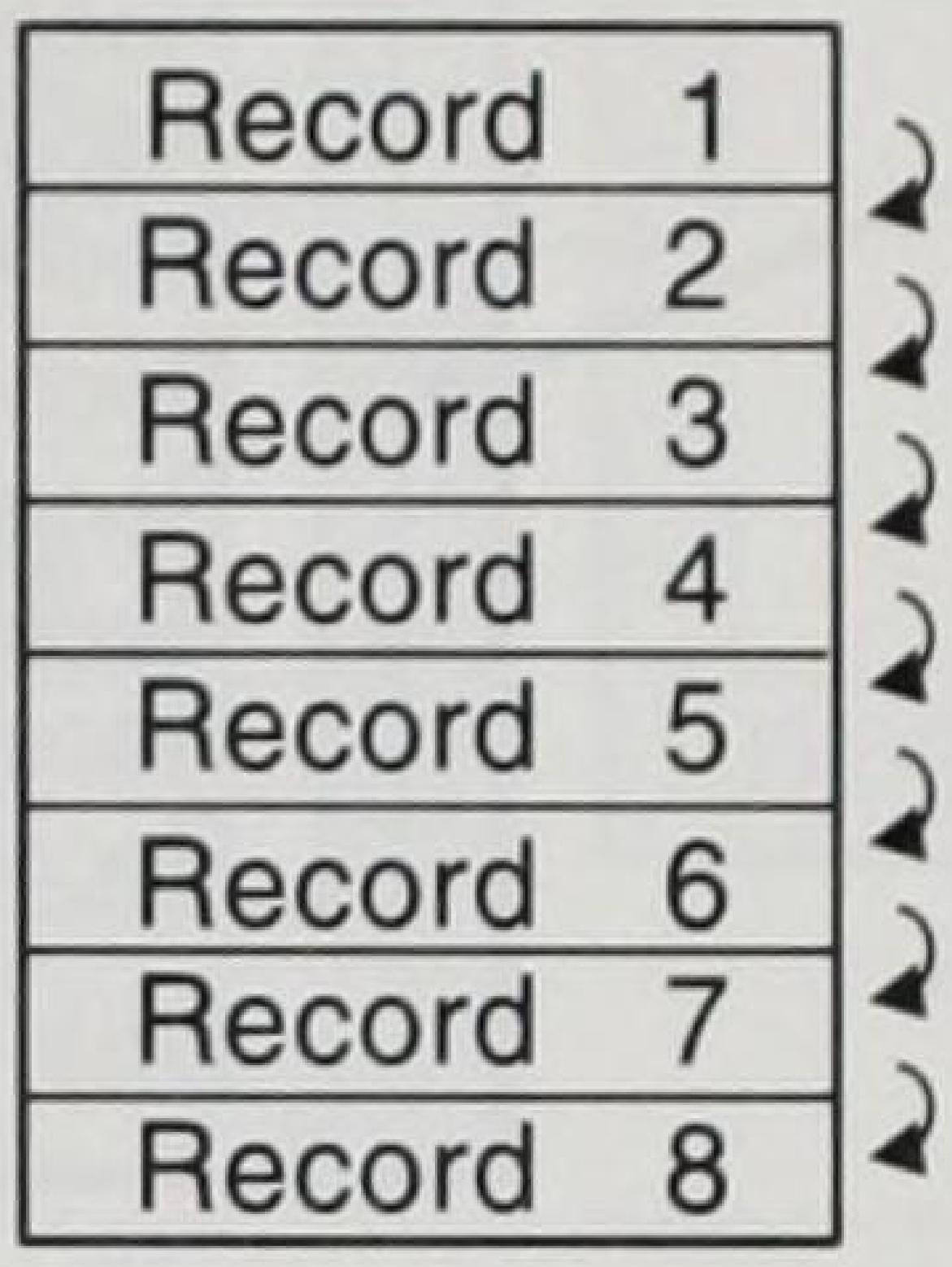
```
BUFFERS=30
```

Depending on your DOS version, you can use from 1 to 99 disk buffers. If you make the number of disk buffers too large, you will actually decrease your system performance for two reasons. First, each time DOS must read information from disk,

System Type	Buffer Recommendation
IBM PC XT (8088)	20
IBM PC XT (80286)	25
80386 computer	30
80486 computer	35

Table 24-2. *Recommended Buffer Settings for Various Computer Types*

DOS first checks to see if the data is present in a disk buffer, as shown in the following illustration:



If the number of disk buffers becomes too large, DOS will spend considerable time simply searching through the disk buffers. Second, each disk buffer requires 528 bytes of memory (512 bytes for the data, and 16 bytes for information DOS uses to locate the buffer). If you specify too many disk buffers, you may use up a considerable amount of memory an application may put to better use.

DOS provides default value settings for different CONFIG.SYS entries as required. Depending on your DOS version and available memory, the amount number of default buffers DOS will differ. In general, use the values shown in Table 24-3 to determine the default buffer settings.

Many applications, such as a database that reads a large number of entries or a word processor that reads a large document, will ask DOS to read one sector that the program will use, followed by a request to read the next sector. As you have learned, DOS stores a file on disk using groups of sectors that form allocation units. If a program asks DOS to read sector 100, for example, it is very likely the program will ask DOS to read sectors 101, 102, and 103.

If you are using DOS 4 or later, you can direct DOS to preread these sectors in anticipation of their later use. DOS refers to the process of prereading sectors as *read-ahead.* Using the CONFIG.SYS BUFFERS entry, you can direct DOS to preread from one to eight sectors. The following CONFIG.SYS entry, for example, directs DOS to use 30 disk buffers to allocate memory for four read-ahead buffers:

```
BUFFERS=30,4
```

System RAM	Default Buffer Settings
128Kb or less	2 for 360Kb disk; 3 for disk larger than 360Kb
129Kb to 255Kb	5
256Kb to 511Kb	10
512Kb to 640Kb	15

Table 24-3. *Default Buffer Settings for Different System Configurations*

Each read-ahead sector consumes a sector's worth of memory (typically 512 bytes). Last, if you are using DOS 4 and your system uses expanded memory, you can include the /X switch to direct DOS to store the buffers in expanded memory, freeing conventional memory for your applications. Chapter 32 discusses memory in detail.

Many users install disk-caching software, which provides more efficient disk buffering than DOS buffers. If you are using disk-caching software such as the Windows or DOS 5 or later SMARTDRV, reduce the number of DOS disk buffers to three.

NOTE

DOS not only uses disk buffers to reduce the number of disk-read operations, but also disk writes. Assume, for example, the database program discussed earlier wants to store the very first 64-byte mailing-list entry to disk. By default, DOS does not write a sector to disk until the sector is full or the program tells DOS to flush all the buffers containing its data to disk. Buffer flushing normally only occurs when the program ends. Therefore, when the database program stores the very first entry, DOS actually just places the entry into an outgoing disk buffer, illustrated in Figure 24-4.

DOS buffers the output data in this way because DOS assumes the program will eventually write more data to that sector. By buffering the output, DOS reduces the number of slow disk writes.

Unfortunately, should you turn off your computer before the program ends, or end the program using CTRL-C, the data in the disk buffer may never actually be recorded on disk. As a result, information you thought you stored is lost. To reduce the chance of DOS not flushing a buffer to disk, only turn off your computer when DOS is displaying its prompt.

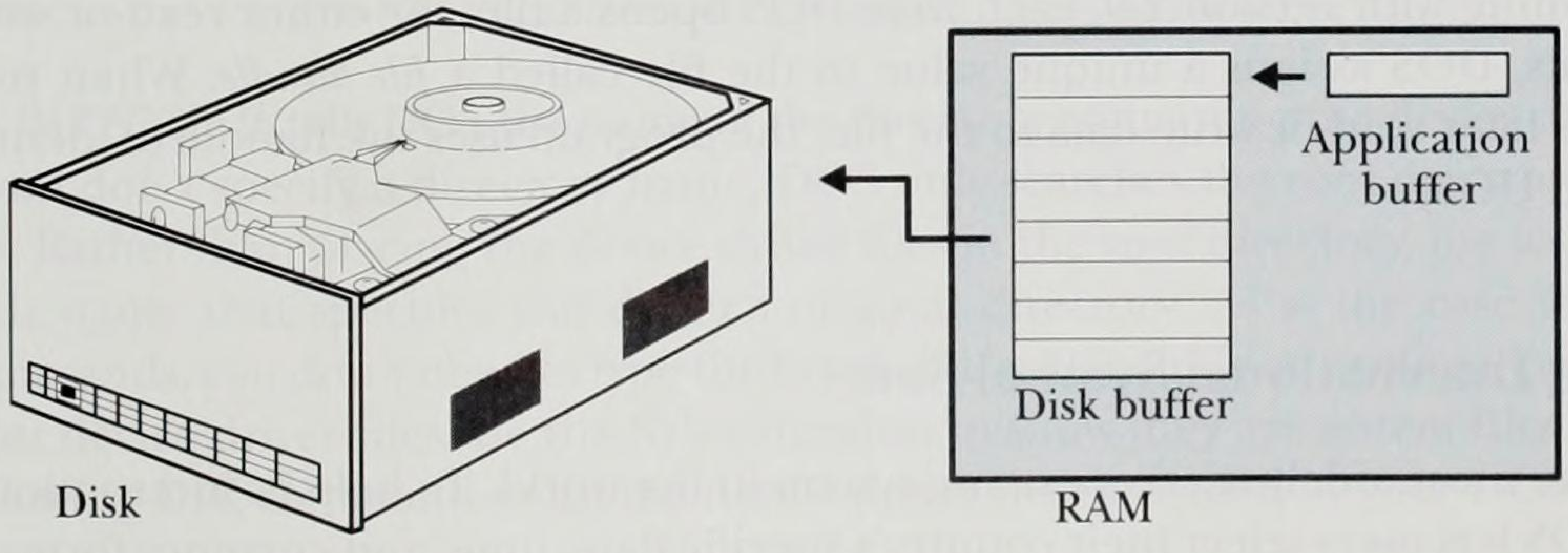


Figure 24-4. *DOS buffers disk output until the buffer is full or the program directs DOS to flush the buffer*

Defining the Number of Files DOS Can Open at One Time

Just as the BUFFERS entry specifies the number of disk buffers DOS can use, the CONFIG.SYS FILES entry specifies the number of files DOS can open at one time. The default value for the FILES entry is 8; but as you will learn, you should use at least 20. Here's why.

In Chapters 20 and 21 you examined the input and output redirection operators. At that time you learned that DOS defines a default source for your program's input called stdin that DOS associates with your keyboard. You also learned that DOS provides default destinations for your command's output (stdout) error messages (stderr), printer output (stdprn), and auxiliary device (stdaux). To provide these input/output source and destinations, DOS automatically opens five files each time you run a program. Since the default setting for the number of files DOS can open is eight, and DOS automatically uses five of them to support I/O redirection, your programs are restricted, by default, to only opening three files. Very few programs can run if they are restricted to three files. If you are using a word processor, for example, the word processor needs to open the file you are editing, possibly a dictionary file used by the spell checker, a file of synonyms used by a built-in thesaurus, and possible font files. If you restrict the word processor to three files, it may not be able to run.

At a minimum, you should provide DOS with the ability to open up to 20 files at the same time using the FILES entry shown here:

```
FILES=20
```

DOS lets you specify up to 255 files. If you are using a large database program or several spreadsheets, the program's installation instructions may direct you to increase the number of files to 40 or 50. Because most programs don't open more files than this at one time, providing DOS with a larger number of handles will probably just waste memory. Each file entry you specify consumes about 64 bytes.

If the number of available files is too low, programs may display error messages telling you they could not open a file you know resides on disk. Should such messages appear, increase the number of available files and reboot your computer.

Beginning with version 2.0, each time DOS opens a file, for either read or write operations, DOS assigns a unique value to the file called a *file handle*. When your programs later read or write data to the file, the program uses the handle to identify the file to DOS.

Specifying International Symbol Sets

DOS is the most widely used operating system in the world. To help its international users, DOS lets users select their country's specific date, time, and currency formats, as well as the character set used to display letters on the screen and printer.

The CONFIG.SYS COUNTRY entry lets international users identify their country to DOS. For more information on the COUNTRY entry, turn to Chapter 38, which examines the capabilities DOS provides international users.

Placing Comments in Your CONFIG.SYS File

In Chapter 15 you used the REM command to place comments (or remarks) in a batch file to explain the batch file's processing. If you are using DOS 4 or later, DOS lets you place REM statements in your CONFIG.SYS file. While processing the CONFIG.SYS entries, DOS ignores any line that begins with REM, continuing with the next entry. Assume, for example, you have a spreadsheet program whose installation instructions direct you to set the FILES entry to 40. Using REM, you can explain the entry, like this:

```
REM Provide file support for Lotus 123 V3.1
FILES=40
```

Should another user examine the contents of your CONFIG.SYS file, REM tells the user the reason for the 40 file handles, reducing the chance that the user will change the entry.

Installing Software Device Drivers

By default, DOS provides software that lets you use your keyboard, screen, printer, and disks. When you add other hardware to your computer such as a mouse, plotter, scanner, or even a built-in FAX, you may need to install additional software that lets DOS communicate with the device. Because such software lets DOS control (or drive) the device, the software is called a *device driver.* To assist DOS, the device driver, like a program, must reside in your computer's memory.

The CONFIG.SYS DEVICE entry lets you install a device driver into memory. The format of the DEVICE entry is as follows:

```
DEVICE=C:\PATHNAME\FILENAME.EXT  [optional_parameters]
```

PATHNAME tells DOS the name of the directory containing the device driver file. If you don't specify a directory name, DOS only searches the root directory for the file. Rather than placing the device driver files in the root directory, use a complete path name that specifies the driver's original directory. As is the case with DOS commands, you don't need to type the brackets if you include an optional parameter. Most device driver files use the SYS extension to show they are system files. Use the following DIR command to list the device driver files provided by your DOS version.

```
C:> DIR  \DOS\*.SYS <ENTER>
```

If you are using DOS 5 or later, DOS also provides the device driver files SETVER.EXE and EMM386.EXE. Likewise, DOS 6 adds the device drivers POWER.EXE, SMARTDRV.EXE, and INTERLNK.EXE.

Table 24-4 defines the device drivers provided by different DOS versions. Most of the device drivers are examined in detail in chapters that best correspond to their use. For now, you need simply understand that a device driver is special software that lets DOS make use of a hardware device.

Driver Name	Function
ANSI.SYS	Provides enhanced keyboard and video support (Chapter 34)
DBLSPACE.SYS	Places the DBLSPACE device driver into conventional memory
DISPLAY.SYS	Provides monitor code-page (character set) switching for international users (Chapter 38)
DRIVER.SYS	Provides software support for external floppy disk drives
EGA.SYS	Provides EGA video support for the DOS 5 or later task swapper (Chapter 40)
EMM386.EXE	Simulates expanded memory in DOS 5 or later systems containing only extended memory (Chapter 32)
HIMEM.SYS	Manages extended memory under DOS 5 or later (Chapter 32)
INTERLNK.EXE	Provides support for file transfers between PCs. Requires DOS 6
POWER.EXE	Provides support for computers with advanced power management chips. Requires DOS 6
PRINTER.SYS	Provides printer code-page (character set) support for international users (Chapter 38)
RAMDRIVE.SYS	Creates a fast electronic RAM drive (Chapter 32)
SETVER.EXE	Defines the version number DOS 5 or later should report when the specific commands ask DOS for the current version number (Chapter 41)
SMARTDRV.SYS	Creates a disk cache in expanded or extended memory (Chapter 32)
VDISK.SYS	Creates a fast electronic RAM drive similar to RAMDRIVE.SYS
XMAEM.SYS	Provides expanded memory support for 80386 machines in PC DOS 4 (Chapter 32)
XMA2EMS.SYS	Provides expanded memory support for 80286 machines in PC DOS 4 (Chapter 32)

Table 24-4. *Device Drivers in Different DOS Versions*

SUMMARY

BUFFERS

The CONFIG.SYS BUFFERS entry lets you specify the number of disk buffers DOS uses to increase your system performance. Compared to your computer's fast electronic parts, even your hard disk is very slow. By using disk buffers, DOS reduces the number of slow disk input and output operations it must perform. The following CONFIG.SYS BUFFERS entry allocates 30 disk buffers:

```
BUFFERS=30
```

FILES

The CONFIG.SYS FILES entry specifies the number of files DOS can open at one time. By default, DOS uses eight file handles, five of which are used for I/O redirection. At a minimum, you should use the FILES entry to specify 20 available files as shown here:

```
FILES=20
```

REM

The CONFIG.SYS REM entry lets you place comments throughout your CONFIG.SYS file that explain the values assigned to each entry. REM requires DOS 4 or later. When DOS encounters REM while processing your CONFIG.SYS entries, DOS ignores the line, continuing its processing at the next entry. The following REM entry explains the BUFFERS entry:

```
REM Use minimal buffers due to caching with SMARTDRV.SYS.
BUFFERS=3
```

DEVICE

The CONFIG.SYS DEVICE entry lets you install a software device driver into your computer's memory. A device driver is special software that lets DOS use a hardware device. The software must reside in your computer's memory before DOS can use it. The following DEVICE entry loads the ANSI.SYS device driver from the DOS directory.

```
DEVICE=C:\DOS\ANSI.SYS
```

Hands On

BUFFERS, FILES, and DEVICE are the most common CONFIG.SYS entries. At a minimum, your CONFIG.SYS file should contain the following entries:

```
FILES=20
BUFFERS=30
```

Use the following DIR command to determine if your root directory has a CONFIG.SYS file:

```
C:\> DIR  \CONFIG.SYS  <ENTER>
```

If the file exists, use TYPE to display its contents:

```
C:\> TYPE  \CONFIG.SYS  <ENTER>
```

If the file does not exist, use your word processor or the DOS editor to create the file using the FILES and BUFFERS entries just shown. Remember, before you change the contents of your CONFIG.SYS file, always save a copy of the existing contents in a file named CONFIG.SAV, as shown in the following listing:

```
C:\> COPY  \CONFIG.SYS  \CONFIG.SAV  <ENTER>
```

Should an added or changed CONFIG.SYS entry result in an error that prevents your computer from booting from hard disk, you can boot your computer using a bootable floppy disk in drive A, and then restore CONFIG.SYS's original contents using the file CONFIG.SAV. In fact, if you have not yet done so, make a copy of your CONFIG.SYS file right now. Also use the PRINT command to print the file's current contents, and place the hardcopy output in a safe location.

```
C:\> PRINT  \CONFIG.SYS  <ENTER>
```

If your current CONFIG.SYS file does not contain the BUFFERS or FILES entry, add the entries previously shown.

Remember, if you add or change CONFIG.SYS entries, you must reboot your computer for the change to take effect. The "Advanced Concepts" section examines the remaining CONFIG.SYS entries.

Review

1. How does CONFIG.SYS differ from AUTOEXEC.BAT?
2. When does DOS use the CONFIG.SYS entries?
3. What happens if your disk doesn't have a CONFIG.SYS file?
4. What is a disk buffer? How do disk buffers improve your system performance?
5. What is the purpose of the CONFIG.SYS FILES entry? Why are DOS programs restricted to only three files if you don't use the FILES entry?

Answers

1. Each time DOS starts, DOS searches your disk's root directory for the files CONFIG.SYS and AUTOEXEC.BAT. The CONFIG.SYS file does not contain commands, but rather, single-line entries that DOS uses to load itself in memory for your specific system configuration. If the file does not exist, DOS uses its own default values. AUTOEXEC.BAT, on the other hand, is a special batch file with commands that DOS automatically executes at system startup. If the AUTOEXEC.BAT file does not exist, DOS invokes the DATE and TIME commands.

2. The only time DOS uses CONFIG.SYS is during system startup. If you make changes to CONFIG.SYS, you must restart your system to put the changes into effect.

3. If your system does not have a CONFIG.SYS file, DOS loads itself in memory using its own default configuration values. If your CONFIG.SYS file does not contain a value for each possible entry, DOS uses its default values.

4. A disk buffer is a region in your computer's fast electronic memory into which DOS reads information from disk. The smallest unit of information DOS can transfer to or from disk is a disk sector. Each time a program asks DOS to read information from disk, DOS reads at least one sector into a disk buffer. When the program asks DOS to read additional information, DOS first checks to see if the information is already in a disk buffer. If the information is in a buffer, DOS can quickly provide the information to the program without having to perform a slow disk read operation. By reducing slow disk reads, disk buffers improve your system performance.

5. The FILES entry defines the number of files DOS can open at one time. The default number is eight. Unfortunately, of these eight files, DOS uses five to support I/O redirection. As a result, unless your CONFIG.SYS file contains a FILES entry, DOS restricts your programs to three files. Most larger programs need more than three files to run.

Advanced Concepts

Although several of the CONFIG.SYS entries that appear in this section are simple to understand, they are presented in the "Advanced Concepts" section because they are less frequently used than the entries previously discussed.

Extended CTRL-BREAK Checking

As you know, DOS lets you end a command by pressing the CTRL-BREAK keyboard combination. When you run a command, DOS periodically checks to see if you have pressed either CTRL-BREAK or CTRL-C to end the command. By default, DOS only performs this checking when it reads a character from the keyboard or when it writes a character to the screen, printer, or auxiliary port. If you have a program that doesn't perform these input or output operations on a regular basis, it may take DOS considerable time to recognize that you have pressed CTRL-BREAK.

Chapter 41 examines the BREAK command that directs DOS to perform extended CTRL-BREAK checking. Using BREAK, the following command turns on extended CTRL-BREAK checking:

```
C:\> BREAK ON <ENTER>
```

When extended CTRL-BREAK checking is on, DOS checks for a user-entered CTRL-BREAK after each DOS system service, such as a disk read or write. Because DOS checks for the CTRL-BREAK more often, DOS is likely to recognize and respond to your CTRL-BREAK sooner. Unfortunately, the amount of extra processing DOS must perform for extended CTRL-BREAK checking makes your system run slower.

As a rule, unless you are a programmer writing and testing programs, you should leave extended CTRL-BREAK checking off.

```
C:\> BREAK OFF <ENTER>
```

By default, DOS starts with extended CTRL-BREAK checking off. If you are a programmer and you want DOS to start with extended CTRL-BREAK checking on, you can place the following BREAK entry in your CONFIG.SYS file.

```
BREAK=ON
```

Specifying the Last Drive Letter

Chapter 26 examines how the SUBST command lets you abbreviate directory names using a logical disk drive letter. By default, DOS lets you use the drive letters A through E. If you create a drive letter greater than E, you must first use the CONFIG.SYS

LASTDRIVE entry to specify the drive letter from A through Z of the last drive letter you intend to use. Chapter 43 examines FDISK, which lets you divide your hard disk into partitions. Using FDISK, you can further divide a disk partition into logical drives. When you create logical drives with FDISK, DOS automatically takes care of correct drive-letter assignments.

The following LASTDRIVE entry selects H as the last drive letter:

```
LASTDRIVE=H
```

If you aren't using logical drives, you don't need the LASTDRIVE entry. If you are using logical drives, only assign LASTDRIVE with the number of letters necessary. Each drive requires 80 to 96 bytes.

Providing File Control Block Support for Older Programs

The CONFIG.SYS FILES entry defines the number of file handles DOS can use to open files. Before version 2.0, DOS did not use file handles, but rather *file control blocks* or *FCBs.* If you are running a very old software program on a network or with the SHARE command loaded, and the program continually displays error messages when it tries to open files, you may need to provide additional file control blocks for the program. The CONFIG.SYS FCBS entry defines the number of file control blocks DOS should provide. The default value is 4. The format of the FCBS entry is as follows:

```
FCBS=MaxOpen,MaxNoClose
```

The *MaxOpen* value specifies the number of file control blocks DOS provides, from 1 to 255. If the older program is experiencing errors, experiment with values of 12 or 16. The *MaxNoClose* value specifies the number of file control blocks SHARE must not close. In some network configurations, DOS experienced problems when many files where opened using file control blocks that were never closed. By default, if four files are open using file control blocks and a program tries to open a fifth, SHARE automatically closes one of the files. The *MaxNoClose* value specifies the number of file control blocks SHARE must leave open. The default setting is 0.

NOTE

File control blocks existed long before disks larger than 32Mb. If you are using DOS 4, the SHARE command provides file control block support for disks larger than 32Mb. For more information on SHARE, see Chapter 30.

Providing Support for Hardware Interrupts

As there are times throughout your day when you need to temporarily put one task aside so you can work on another, the same is true for DOS. When a device such as your disk finishes an I/O operation, the device interrupts DOS to request immediate attention. DOS in turn, puts aside its current task so it can handle the *device interrupt.*

Because DOS must be able to later resume the task, DOS stores information about the first task in a memory location called a *stack*. Each time DOS receives an interrupt, DOS allocates one stack from the available stacks. Depending on your computer type and hardware, there may be times when so many interrupts occur in such a short period of time that DOS runs out of stack space. Should this occur, your system will halt, displaying the following error message:

```
Fatal: Internal Stack Failure, System Halted
```

If DOS runs out of stack space in this way, place a STACKS entry in your CONFIG.SYS. The STACKS entry directs DOS to allocate additional memory to support hardware interrupts. The format of the STACKS entry is as follows:

```
STACKS=NumberofStacks,StackSize
```

The only time you need to include the STACKS entry in CONFIG.SYS is when your computer hangs, displaying the "Fatal stacks" error. If this error occurs, experiment with different stack settings, beginning with

```
STACKS=8,512
```

Table 24-5 defines the number of stacks DOS uses by default for different computer types.

Try to keep the stack size as small as possible. The previous entry uses 4096 bytes (8 × 512 = 4096).

Installing Memory-Resident Commands

Several DOS commands install memory-resident software. If you are using DOS 4 or later and the FASTOPEN, KEYB, NLSFUNC, or SHARE commands, you should use

Computer Type	STACKS Entry
IBM PC, PC XT, Portables	0,0
All others	9,128

Table 24-5. Default STACKS Entries

the CONFIG.SYS INSTALL entry to load the commands to make more efficient use of your computer's memory. The format of the INSTALL entry is as follows:

```
INSTALL=CommandName [optional_parameters]
```

For example, the following entry loads the FASTOPEN command:

```
INSTALL=C:\DOS\FASTOPEN.EXE  C:=50
```

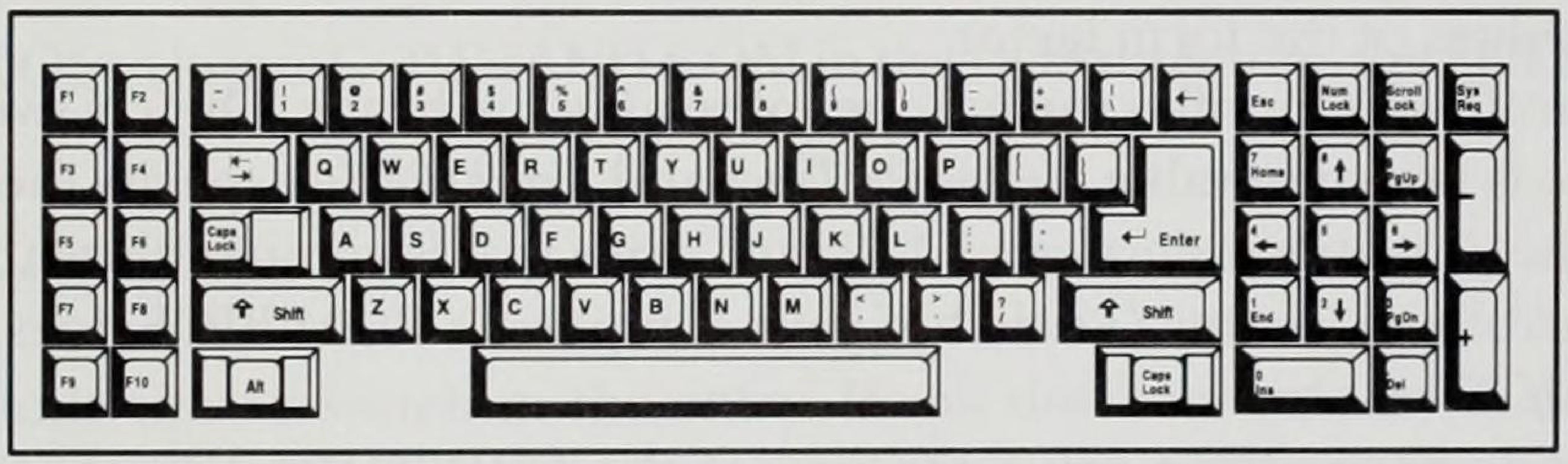
Although the INSTALL entry makes efficient use of your computer's memory, it only supports these four DOS commands. Do not use INSTALL to load other memory-resident commands.

Making an Enhanced Keyboard Behave as a Conventional Keyboard

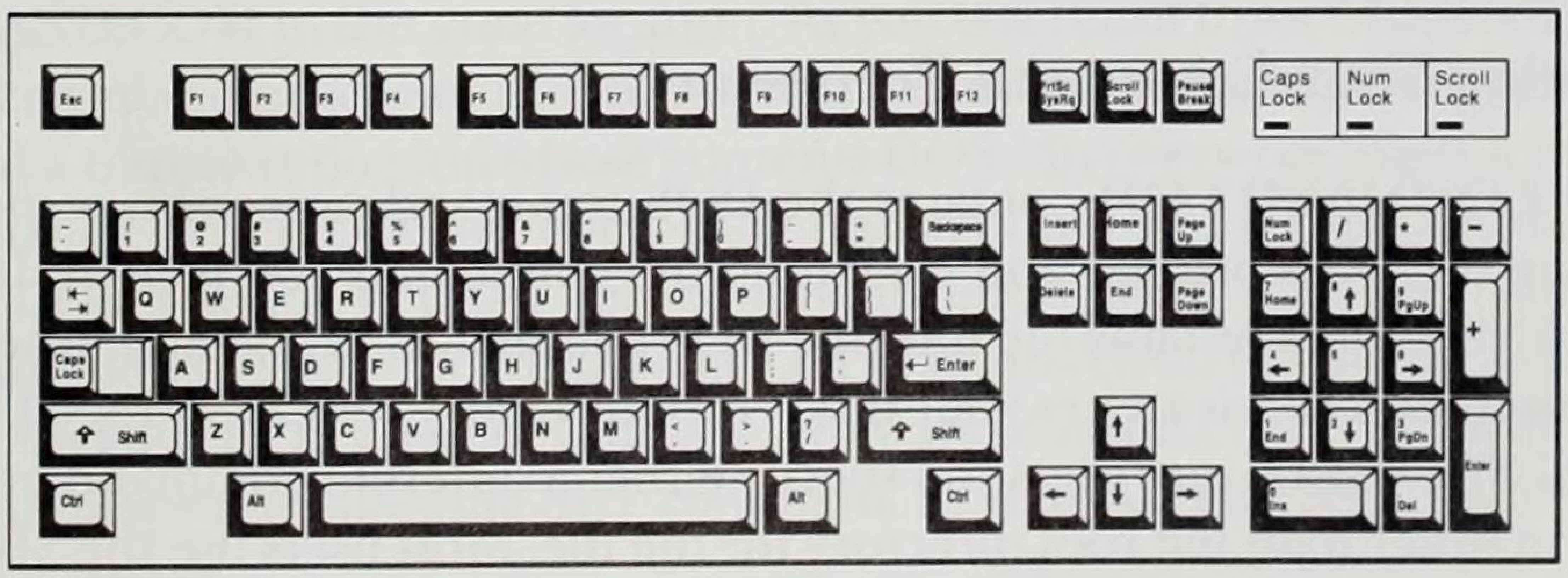
Figure 24-5 illustrates a conventional and an enhanced keyboard.

If you are running an older program, the program may not support several of the enhanced keyboard's keys. In DOS 4 and later the CONFIG.SYS SWITCHES entry lets you direct DOS to treat the enhanced keyboard as a conventional keyboard.

```
SWITCHES=/K
```


F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
Esc
Num Lock
Scroll Lock
Sys Req
Q
W
E
R
T
Y
U
I
O
P
Home
PgUp
Caps Lock
A
S
D
F
G
H
J
K
L
Enter
Shift
Z
X
C
V
B
N
M
Shift
End
PgDn
Alt
Caps Lock
Ins
Del

Standard keyboard


Esc
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
F11
F12
PrtSc SysRq
Scroll Lock
Pause Break
Caps Lock
Num Lock
Scroll Lock
Backspace
Insert
Home
Page Up
Num Lock
Q
W
E
R
T
Y
U
I
O
P
Delete
End
Page Down
Home
PgUp
Caps Lock
A
S
D
F
G
H
J
K
L
Enter
Shift
Z
X
C
V
B
N
M
Shift
End
PgDn
Enter
Ctrl
Alt
Alt
Ctrl
Ins
Del

Enhanced keyboard

Figure 24-5. *Standard and enhanced keyboards*

If you are using the ANSI.SYS device driver and SWITCHES /K, install the ANSI.SYS driver using the /K switch.

Changing Physical Drive Parameters

DOS supports two types of devices, character devices and block devices. Devices are classified as one type or the other, depending on how DOS sends or receives data to or from the device. *Character devices* such as your printer receive information a byte or character at a time. *Block devices* such as your disk receive information in groups (or blocks) of bytes. Beginning with DOS 3.2, the CONFIG.SYS DRIVPARM entry lets you change one or more block device characteristics. Most users won't need to use DRIVPARM; however, using DRIVPARM, you may be able to change the characteristics of a drive to read a disk containing data in a unique format.

The format of the DRIVPARM entry is

```
DRIVPARM=/D:DriveNumber [/C] [/F:FormFactor] [/H:NumberofHeads]
    [/N] [/S:SectorsPerTrack] [/T:TracksPerSide] [/I]
```

DriveNumber specifies the drive's physical number from 0 through 25, where 0 corresponds to drive A, 1 to drive B, and 2 to C. /C indicates the drive provides change-line support. Each time you open and close a floppy disk drive, most floppy drives provide a signal to DOS telling DOS to check for a possible disk change. This signal is called a *change line.*

FormFactor is a unique value that specifies the device type. Table 24-6 defines the possible values of the form factor.

NumberofHeads specifies the number of read/write heads. The /N switch indicates the device is nonremovable. *SectorsPerTrack* and *TracksPerSide* specify the disk's sector layout. The /I switch indicates a 3 1/2-inch drive that is compatible with the existing drive controller. Use the /I switch if your computer's ROM-BIOS does not support 3 1/2-inch disks.

If you don't specify a value for one of the DRIVPARM entries, DOS uses the device's existing value.

Customizing Your Command Processor

The file COMMAND.COM contains the DOS command processor, responsible for displaying the DOS prompt and executing the commands you type in. In addition, COMMAND.COM contains the internal DOS commands. By default, when you start your computer, DOS searches your disk's root directory for COMMAND.COM. The CONFIG.SYS SHELL entry lets you specify either a different command processor, or a location other than the root directory for the file. Most users use the SHELL entry to direct DOS to find COMMAND.COM in the DOS directory, so they can delete

Form Factor	Device Type
0	160Kb, 180Kb, 320Kb, or 360Kb floppy drive
1	1.2Mb floppy drive
2	720Kb floppy drive
3	8-inch single-density floppy drive
4	8-inch double-density floppy drive
5	Hard drive
6	Tape drive
7	1.44Mb floppy drive
8	Optical drive (DOS 5 or later)
9	2.88Mb floppy drive (DOS 5 or later)

Table 24-6. *DRIVPARM Form Factor Values*

COMMAND.COM from the root directory. The format of the SHELL entry is as follows:

```
SHELL=\PATHNAME\FILENAME.EXT  [optional_parameters]
```

To direct DOS to locate COMMAND.COM in the DOS directory, the SHELL entry becomes

```
SHELL=C:\DOS\COMMAND.COM  /P
```

You must include the /P switch in the entry. If you don't include it, DOS will not execute the AUTOEXEC.BAT batch file or the DATE and TIME commands.

You should note that if you use SHELL to locate COMMAND.COM in a directory other than the root, you must also update the COMSPEC environment entry to look for COMMAND.COM in the same location. As you will recall from Chapter 19, DOS divides the command processor into two parts: a resident portion that always remains in RAM, and a transient portion whose contents DOS can overwrite when it executes large commands. When DOS overwrites the transient portion during a command's execution, DOS reloads it when the command ends. DOS uses the COMSPEC environment entry to locate COMMAND.COM so it can reload the transient portion. Depending on your DOS version, you have two ways of setting the COMSPEC entry. First, you can place the following SET command in your AUTOEXEC.BAT file:

```
SET COMSPEC=C:\DOS\COMMAND.COM
```

Depending on your DOS version, you can include COMMAND.COM's directory location in the SHELL entry,

```
SHELL=C:\DOS\COMMAND.COM  C:\DOS  /P
```

and DOS will automatically set the COMSPEC entry for you.

Increasing Your Environment Size

As you learned in Chapter 19, DOS initially allocates 160 bytes of memory for the environment. If you fill the environment, and have not installed memory-resident software, DOS will increase the environment size as needed. If DOS is unable to increase the environment size, it will display the following error message:

```
Out of environment space
```

Using the SHELL entry, you can increase the environment's starting size. The following entry reserves a 512-byte environment:

```
SHELL=C:\DOS\COMMAND.COM  /P  /E:512
```

DOS lets you use the /E switch to specify an environment size from 160 bytes to 32,768 bytes.

Loading Error Messages into Memory

If you are using DOS 4 or later and running DOS from floppy disk, you may want to include the /MSG switch with your SHELL entry, which directs DOS to load its error messages into RAM.

By default, to reduce its memory use, DOS stores many seldom-used error messages on disk. Should DOS need to display one of those messages, DOS simply reads the message from disk. If you are using floppy disks, however, it may not be convenient for DOS to always look to disk to find these messages. The following SHELL entry uses /MSG to load the DOS error messages into RAM:

```
SHELL=COMMAND.COM  /P  /MSG
```

Memory Management in DOS 5 or later

If you are using DOS 5 or later, the CONFIG.SYS DOS and DEVICEHIGH entries let you make very efficient use of your computer's conventional memory. Before you can fully understand how these entries work, you must understand conventional, reserved, and high memory areas. Chapter 32 examines DOS memory management in detail. The CONFIG.SYS DOS and DEVICEHIGH entries are presented in more detail in that chapter.

SUMMARY

BREAK

The CONFIG.SYS BREAK entry lets you enable extended CTRL-BREAK checking at system startup. By default, DOS starts with extended CTRL-BREAK checking off. The BREAK command, presented in detail in Chapter 41, lets you turn extended CTRL-BREAK checking on and off as needed. The CONFIG.SYS entry selects the starting mode, on or off.

LASTDRIVE

The CONFIG.SYS LASTDRIVE entry lets you specify the drive letter of the last logical disk drive DOS is to recognize. The FDISK and SUBST commands create logical drives. The following LASTDRIVE entry directs DOS to recognize drives A through J:

```
LASTDRIVE=J
```

FCBS

The CONFIG.SYS FCBS entry provides support for older programs (typically before DOS 2) that use file control blocks to perform file input/output operations. If you are not using very old programs on a network or with SHARE active, you won't need to use the CONFIG.SYS FCBS entry. If an older program experiences file errors, set the number of file control blocks to 12, as shown here:

```
FCBS=12
```

STACKS

The CONFIG.SYS STACKS entry directs DOS to allocate additional memory to handle hardware interrupts. The only time you need to use the STACKS entry is when your system hangs and displays the following message:

```
Fatal: Internal Stack Failure, System Halted
```

Should this error message occur, experiment with the following entry:

```
STACKS=8,512
```

INSTALL

In DOS 4 or later, the CONFIG.SYS INSTALL entry lets you load the memory-resident DOS commands KEYB, NLSFUNC, FASTOPEN, and

SUMMARY

SHARE in a more efficient manner than if the commands were simply called. The following INSTALL command loads the SHARE command:

```
INSTALL=C:\DOS\SHARE.EXE
```

DRIVPARM

The CONFIG.SYS DRIVPARM entry lets you change the characteristics of a block device such as a disk or tape drive. Most users won't require DRIVPARM. The following entry configures a tape drive installed as drive E (physical device number 4):

```
DRIVPARM=/D:4  /H:1  /F:6  /T:20
```

SHELL

The CONFIG.SYS SHELL entry lets you specify the DOS command processor. By default, DOS uses the file COMMAND.COM, which resides in your disk's root directory. To eliminate duplicate copies of COMMAND.COM from their disk, many users use the SHELL entry to direct DOS to locate COMMAND.COM in the DOS directory. The following SHELL entry directs DOS to locate COMMAND.COM in the DOS directory:

```
SHELL=C:\DOS\COMMAND.COM /P
```

If you move COMMAND.COM from the root in this way, you must use the following SET command in your AUTOEXEC.BAT file to set the COMSPEC environment entry:

```
SET  COMSPEC=C:\DOS\COMMAND.COM
```

The following SHELL entry includes the C:\DOS directory to automatically set the COMSPEC environment entry to the same:

```
SHELL=C:\DOS\COMMAND.COM  C:\DOS  /P
```

DOS 6 System Configuration Options

Many users have different CONFIG.SYS files that they use for different types of processing. For example, if they are going to use a network, they use the CONFIG.SYS file that contains the network device drivers. Likewise, if they need to free up conventional memory for a large program, they use a second file. DOS 6, however, provides CONFIG.SYS entries that let you customize your system startup. In fact, you can create a menu of options from which you can select the desired configuration. Table 24-7 lists the DOS 6 CONFIG.SYS customization entries.

To perform customized startup processing, you divide up your CONFIG.SYS file into groups of entries. For example, you would group the entries you use for a network configuration in one section and those you use without the network in another. Each block (or section) has a unique name that can contain up to 70 characters. You place the block name within left and right brackets: [*Name*]. Regardless of the different configurations you use, you will normally have several entries that are common to each group, such as your FILES, BUFFERS, or STACKS entries. Therefore, DOS provides a special block named [COMMON] within which you can place such entries. The following CONFIG.SYS file contains three blocks, one for a network configuration, one for no network, and one for common entries.

```
[COMMON]
FILES=50
BUFFERS=30
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[NETWORK]
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE

[NoNETWORK]
DEVICEHIGH=RAMDRIVE.SYS 512
DEVICEHIGH=ANSI.SYS
```

Entry	Purpose
MENUCOLOR	Specifies the foreground and background color of the startup menu
MENUITEM	Specifies a system configuration menu option
MENUDEFAULT	Specifies the startup menu's default option
INCLUDE	Includes the entries contained in the specified configuration block
SUBMENU	Specifies a lower-level menu of startup options

Table 24-7. *DOS 6 CONFIG.SYS System Configuration Entries*

Next, you need a way to select which configuration you desire. To do so, you simply create a menu of options using the CONFIG.SYS MENUITEM entries. To use MENUITEM, you specify the block name and optional menu text as shown.

MENUITEM=*BlockName*[,*MenuText*]

For example, the following entries create options for the network and no-network configurations:

```
[MENU]
MENUITEM=NETWORK,Network configuration
MENUITEM=NoNETWORK,No network support
```

When you combine these entries with the previously defined configuration blocks, the CONFIG.SYS file becomes the following:

```
[MENU]
MENUITEM=NETWORK,Network configuration
MENUITEM=NoNETWORK,No network support

[COMMON]
FILES=50
BUFFERS=30
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[NETWORK]
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE

[NoNETWORK]
DEVICEHIGH=RAMDRIVE.SYS 512
DEVICEHIGH=ANSI.SYS
```

Note that the [COMMON] block is special. DOS performs the entries listed in the [COMMON] block regardless of the configuration option you select.

Assuming you restart the system containing this CONFIG.SYS file, DOS will display the following menu on your screen:

```
MS-DOS 6 Startup Menu
```

```
     1. Network configuration
     2. No network support
Enter a choice: 1
```

To select an option, type the option's number or highlight the option using your arrow keys and press ENTER.

If you look closely at the format of the MENUITEM entry you will find the menu text is optional. If you don't specify options text, DOS will display the block name in the startup menu. For example, the menu shown in the following does not specify any menu option text.

```
[MENU]
MENUITEM=NETWORK
MENUITEM=NoNETWORK
```

When your system starts, DOS will display the following:

```
MS-DOS 6 Startup Menu

    1. NETWORK
    2. NoNETWORK
Enter a choice: 1
```

Specifying a Menu Color

To help you further customize your system startup, the CONFIG.SYS MENUCOLOR entry lets you specify the color of the startup menu. The format of the entry is as follows.

MENUCOLOR=*Foreground*[,*Background*]

The foreground and background colors are specified using a color value from 0 through 15 as listed in Table 24-8.

The following entry directs DOS to display bright white menu text on blue background:

```
[MENU]
MENUCOLOR=15,1
MENUITEM=NETWORK,Network configuration
MENUITEM=NoNETWORK,No network configuration
```

Color Value	Color	Color Value	Color
0	Black	8	Gray
1	Blue	9	Bright blue
2	Green	10	Bright green
3	Cyan	11	Bright cyan
4	Red	12	Bright red
5	Magenta	13	Bright magenta
6	Brown	14	Yellow
7	White	15	Bright white

Table 24-8. *CONFIG.SYS MENUCOLOR Color Values*

Selecting a Default Menu Option

If you examine the startup screens displayed for the previous menu configurations, you will find that DOS lists option 1 as the default. Depending on your configuration, you may want to specify a different option as the default. The CONFIG.SYS MENUDEFAULT option lets you specify the default menu option. In addition, MENUDEFAULT lets you specify a timeout period from 0 to 90 seconds after which DOS automatically selects the option. By specifying a time-out option, you can ensure that your system starts, even if you are away from the keyboard. The format of the MENUDEFAULT entry is as follows:

MENUDEFAULT=*BlockName*[,*Timeout*]

The following MENUDEFAULT entry directs DOS to select the NETWORK configuration after 30 seconds:

```
[MENU]
MENUITEM=NETWORK,Network configuration
MENUITEM=NoNETWORK,No network support
MENUDEFAULT=NETWORK,30
```

When you specify a default option and timeout, DOS will display a countdown of the number of seconds remaining until it selects the option as shown here:

```
MS-DOS 6 Startup Menu
   1. Network configuration
   2. No network support

Enter a choice: 1                         Time remaining: 15
```

Including One Block's Options Within Another

As you have learned, to configure CONFIG.SYS, you divide the file into multiple blocks. Depending on how you divide your entries, there may be times when one of your blocks contains entries defined in another. Rather than duplicating the entries, you can use the INCLUDE entry to include the second block's entries, as shown here:

INCLUDE=*BlockName*

For example, assume that your CONFIG.SYS file contains a block that is called [HIGH_MEMORY], as shown here:

```
[HIGH_MEMORY]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
```

Within a block such as [NETWORK], you can include the [HIGH_MEMORY] entries, as shown here:

```
[MENU]
MENUCOLOR=15,1
MENUITEM=NETWORK,Network configuration
MENUITEM=NoNETWORK,No network configuration

[COMMON]
FILES=50
BUFFERS=30

[HIGH_MEMORY]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
```

```
[NETWORK]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE

[NoNETWORK]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 512 /E
DEVICEHIGH=C:\DOS\ANSI.SYS
```

Creating Several Menu Levels

Depending on your system configuration, there may be times when you can't easily express the different configuration options using just one menu. Therefore, the SUBMENU entry lets you specify additional menus. The format of the SUBMENU entry is as follows:

SUBMENU=*BlockName*[,*MenuText*]

The block name specifies a CONFIG.SYS block that contains additional MENUITEM entries. When your system starts, DOS will find the [MENU] block and process the entries it contains. If you select an entry that corresponds to a submenu, DOS will display a second menu, followed possibly by a third or fourth. Assume, for example, that if the user selects the "No network support" option, you want to further prompt the user for how they want to load DOS into memory. The following CONFIG.SYS file uses the SUBMENU entry to do just that:

```
[MENU]
MENUCOLOR=15,1
MENUITEM=NETWORK,Network configuration
SUBMENU=NoNETWORK,No network configuration

[COMMON]
FILES=50
BUFFERS=30

[HIGH_MEMORY]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[NETWORK]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE
```

```
[NoNETWORK]
MENUITEM=DOS_HIGH,Load DOS High and use UMB
MENUITEM=DOS_LOW,Load DOS Low without UMB
MENUITEM=DOS_UMB,Load DOS Low with UMB

[DOS_HIGH]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 512 /E
DEVICEHIGH=C:\DOS\ANSI.SYS

[DOS_LOW]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\RAMDRIVE.SYS 512 /E
DEVICE=C:\DOS\ANSI.SYS

[DOS_UMB]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\EMM386.EXE NOEMS
DOS=LOW,UMB
DEVICEHIGH=C:\RAMDRIVE.SYS 512 /E
DEVICEHIGH=C:\ANSI.SYS
```

Further Controlling Your System Startup

In addition to letting you create menu-driven, customized startups, DOS 6 lets you bypass the CONFIG.SYS and AUTOEXEC.BAT processing altogether or selectively perform or ignore each CONFIG.SYS entry. Each time your system starts, DOS 6 displays the following message:

```
Starting MS-DOS...
```

If you press the F5 key, DOS will bypass your CONFIG.SYS and AUTOEXEC.BAT processing, displaying its prompt. Bypassing the startup in this way is useful when one or more entries is preventing DOS from successfully starting your system.

If you instead press the F8 key, DOS will individually prompt you for each CONFIG.SYS entry as to whether or not you want to perform the entry. For example, the following prompt asks whether or not you want to install the ANSI.SYS device driver.

```
DEVICE=C:\DOS\ANSI.SYS [Y,N]?
```

If you type **Y**, DOS will install the driver. If you instead type **N**, DOS will not install the driver. In either case, DOS will continue the same processing with the next entry.

Using SET Within CONFIG.SYS

As discussed in Chapter 19, the DOS SET command lets you assign environment entries. If you are using DOS 6, you can use SET from within your CONFIG.SYS file. For example, assume that you have commands in your AUTOEXEC.BAT file that are dependent on a network or no-network configurations. Using SET within CONFIG.SYS, you can assign a value to the NETWORK entry as shown here:

```
[NETWORK]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE
SET NETWORK=YES

[NoNETWORK]
INCLUDE=HIGH_MEMORY
DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 512 /E
DEVICEHIGH=C:\DOS\ANSI.SYS
SET NETWORK=NO
```

Within your AUTOEXEC.BAT, you can use the named parameter %NETWORK% to test for the network configuration, as shown here:

```
IF %NETWORK%==YES NET_COMMAND
```

Key Terms

Buffer A storage location. A disk buffer is a storage location in memory into which DOS reads disk sectors.

Read-ahead The process of reading the next disk sector in anticipation of its use. In DOS 4 or later, the CONFIG.SYS BUFFERS entry lets you specify the number of read-ahead buffers.

File handle A unique value (assigned to a file each time it is opened by DOS 2 or later) that is used to identify a file to DOS.

File control block A collection of information about an open file. Prior to DOS 2, DOS used file control blocks (FCBs) to access files. Starting with version 2.0, DOS uses file handles.

Stack A location often used to store the computer's status so DOS can handle a device interrupt. Stacks are so named because should several interrupts occur in succession, DOS stacks its status on top of the previous status (like a stack of cafeteria trays). As DOS completes one interrupt, DOS removes the top item (status information) off the stack.

Chapter 25

Using CHKDSK

The more you use your computer, the more information you will store on your hard disk. This information may contain key budget data you produced with your spreadsheet or important reports or memos you created with your word processor. Should your keyboard, printer, or monitor quit working, you can replace it and quickly resume your work. If your disk fails, however, depending on the state of your file backups, you could lose a great deal of information.

Your hard disk is a mechanical device that will eventually wear out. In most cases, DOS will give you hints (in the form of disk read or write error messages) of your disk's impending doom. Rather than sit and wait for such error messages to appear, you can use the CHKDSK (check disk) command to examine your disk's current health. If CHKDSK encounters disk errors or inconsistencies, it will display explanatory error messages on your screen. In many cases, CHKDSK can repair the error. In other cases, you may need to back up the files on the disk to floppies, format the disk, and then restore the files. Get into the habit of invoking CHKDSK on a regular basis to examine your disk.

Understanding CHKDSK

CHKDSK is an external DOS command that checks a disk's status. To examine the current drive, invoke CHKDSK as shown in the following example.

```
C:\> CHKDSK <ENTER>
```

To examine a drive other than the default, include the drive letter in CHKDSK's command line. The following CHKDSK command, for example, examines the floppy disk in drive A:

```
C:\> CHKDSK  A: <ENTER>
```

Depending on your DOS version, the information CHKDSK displays may differ. In DOS 3.3, for example, CHKDSK will display something similar to the following:

```
Volume DOS         created Jul 24, 1993 8:23p

 21309440 bytes total disk space
    53248 bytes in 3 hidden files
    47104 bytes in 12 directories
  4980736 bytes in 363 user files
    55296 bytes in bad sectors
 16173056 bytes available on disk

   655360 bytes total memory
   527488 bytes free
```

In DOS 4 and later, CHKDSK includes additional information describing the disk's allocation units, as shown here:

```
Volume DOS         created 07-24-1993 10:08a
Volume Serial Number is 1661-AB75

 200065024 bytes total disk space
     81920 bytes in 3 hidden files
    450560 bytes in 105 directories
  41267200 bytes in 1894 user files
 158261248 bytes available on disk

      4096 bytes in each allocation unit
     48844 total allocation units on disk
     38638 available allocation units on disk
    655360 total bytes memory
    510080 bytes free
```

As you learned in Chapter 7, the FORMAT command marks damaged sectors as unusable. If your disk contains damaged sectors, CHKDSK will include a line describing the number of bytes of bad sectors your disk contains.

```
1213952 bytes total disk space
868864 bytes in 27 user files
177152 bytes in bad sectors
167936 bytes available on disk

655360 total bytes memory
559776 bytes free
```

If CHKDSK discovers errors or inconsistencies, it will display a line similar to the following, where *n* is the actual number:

```
Errors found, F parameter not specified
Corrections will not be written to disk

n lost allocation units found in nn chains.
nnnn bytes disk space would be freed
```

If CHKDSK displays this error message, you will need to invoke CHKDSK a second time using the /F switch to correct the error, as discussed later in this chapter.

For now, examine each line of CHKDSK's output in detail.

Volume *VOLNAME* created *mm-dd-yy HH:MM*

A volume label is a name you assign to your disk when you format the disk or use the LABEL command (discussed in Chapter 30). If you have assigned a label to your disk, CHKDSK will display the label name and the date and time you assigned the label, for example:

```
Volume DOS          created 07-24-1993 10:08a
```

Volume Serial Number is *xxxx-xxxx*

If you are using DOS 4 or later, the FORMAT command assigns a unique serial number to your disk. Neither DOS nor any other application actually uses the serial numbers yet. However, for DOS 4 or later, CHKDSK, like DIR, will display the disk's serial number, if it exists.

```
Volume Serial Number is 1661-AB75
```

nnnnnn bytes total disk space

This line tells you the disk's total space, including unusable damaged sectors. When you buy a hard disk, the dealer describes your disk size in megabytes. A megabyte (Mb) is 1,048,576 bytes. If you have a 30Mb disk, for example, you would expect your disk to have 31,457,280 total bytes. As it turns out, hard disks often contain more total bytes than guaranteed, which means it may have a megabyte or more than you expect when you actually format the disk.

nnnn bytes in *n* hidden files

For your computer to start DOS from your hard disk, your disk must contain the two hidden system files that were discussed in Chapter 7. In Chapter 23, you learned that in DOS 5 or later, the ATTRIB command lets you hide files yourself for privacy and increased protection. This line of output describes the number of hidden files on your disk and the amount of disk space the hidden files consume. If you have assigned a volume name to your disk using FORMAT or the LABEL command (Chapter 30), DOS stores the volume name as a hidden file in the root directory, which may explain why CHKDSK displays three hidden files, as opposed to two.

```
81920 bytes in 3 hidden files
```

nnnnn bytes in *nn* directories

A directory is a list of filenames, extensions, sizes, and date and time stamps. DOS stores directories on your disk. This output line tells you the number of directories your disk contains and the amount of disk space DOS must use to store the directory lists. As you will learn in Chapter 33, every file entry in a directory requires 32 bytes.

nnnnnn bytes in *nnn* user files

This line tells you the number of files on your hard disk and the amount of disk space the files consume. This does not include hidden files, whose contents CHKDSK has previously taken into account.

nnnn bytes in bad sectors

The FORMAT command marks damaged sectors as unusable to prevent DOS from trying to store information in the damaged disk locations. For older hard disks, bad sectors consuming 1 to 2 percent of the disk's space were not uncommon. Newer disks, however, typically format with no damaged sectors. If your disk's damaged sectors are less than 1 to 2 percent of your total space, don't be too concerned.

nnnnnn bytes available on disk

This line tells you the amount of disk space still available for use on your disk. If you subtract from the total disk space the number of bytes used for hidden files, directories, user files, and damaged sectors, the result should equal the amount of available disk space.

nnnnnn total bytes memory

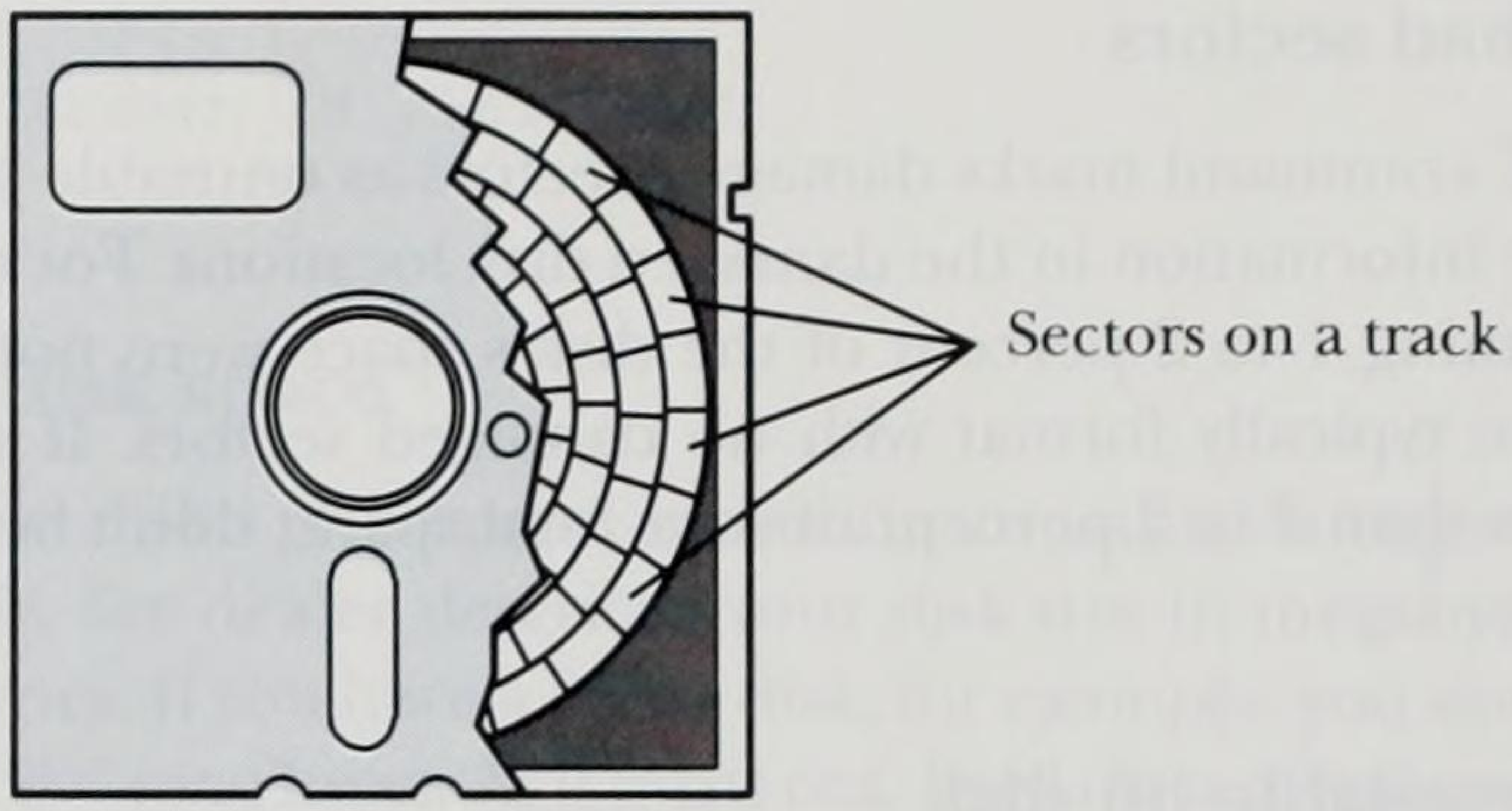
As you know, your computer uses its electronic memory to run programs and uses your disk to store information from one user session to the next. This line describes the amount of conventional electronic memory your computer contains. Chapter 32 examines the different memory types: conventional, extended, and expanded memory, for example. CHKDSK does not display statistics for these memory types. If you are using DOS 4 or later, you can display such information using the MEM command.

nnnnnn bytes free

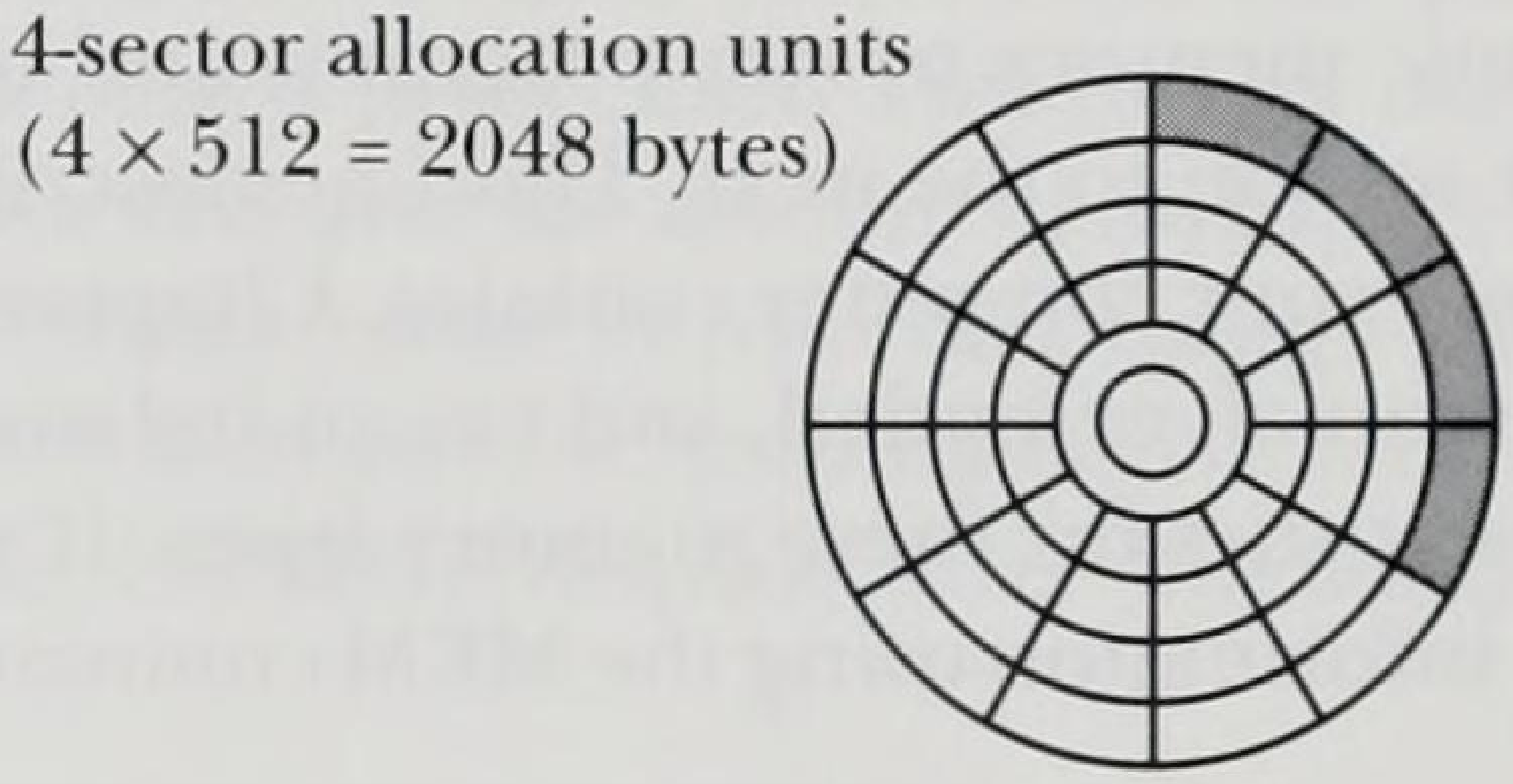
This line tells you how much of your computer's electronic memory is available for program use. As you know, each time you start your computer, it loads DOS from disk into its electronic memory. Subtracting this line's value from the total memory tells you how much memory DOS and any other memory-resident programs you have loaded consume. Many applications will specify a minimum amount of memory they need to run. Using CHKDSK, you can ensure that you have enough available memory.

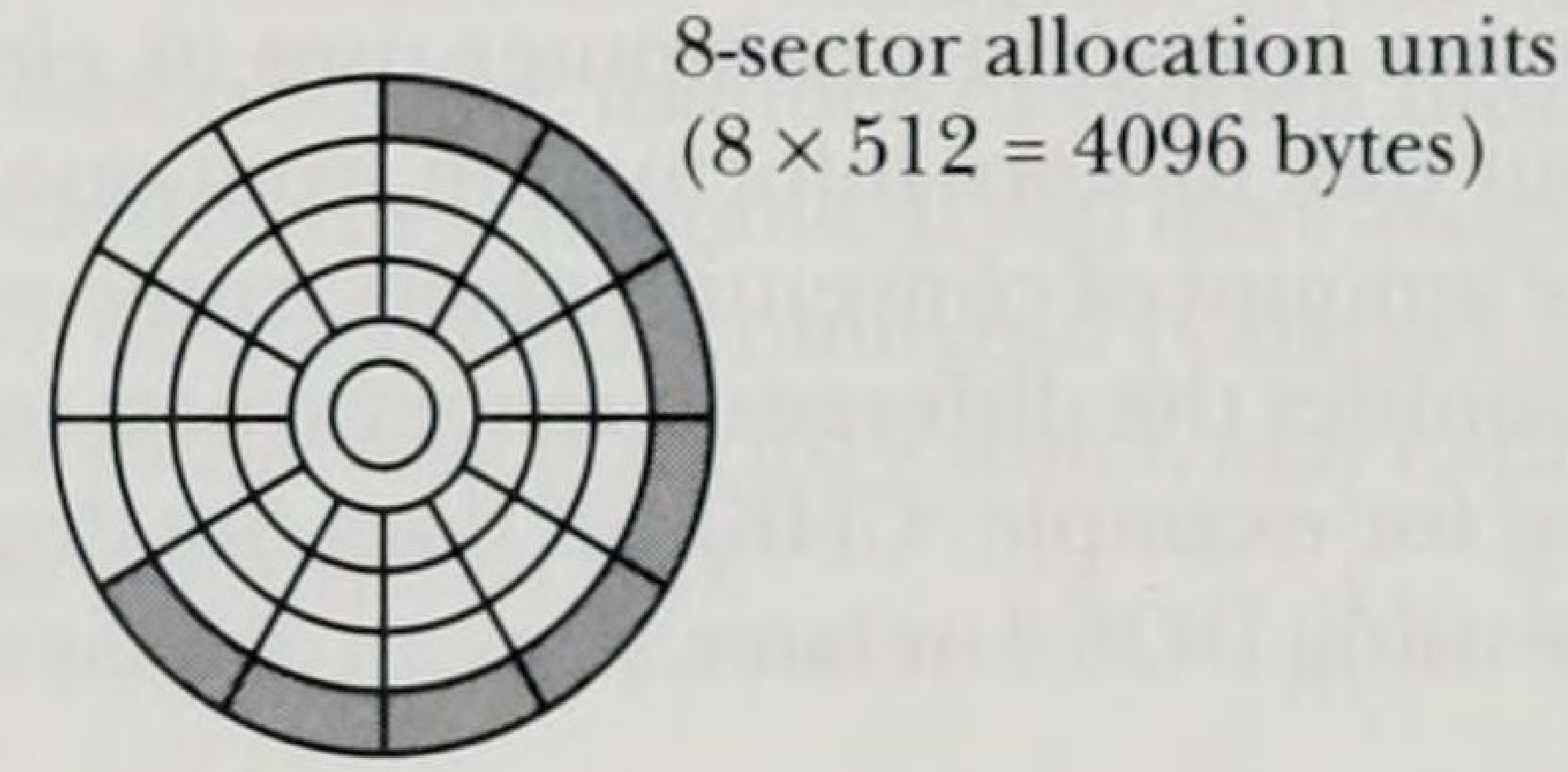
nnnn bytes in each allocation unit

If you are using DOS 4 or later, CHKDSK displays information about your disk's allocation units (clusters). As you learned in Chapter 7, FORMAT first divides your disk into concentric tracks, then further divides each track into storage units called sectors, as illustrated here:



DOS stores information on your disk by recording the information in the disk sectors. For simplicity, users normally assume DOS stores a file's contents on disk a sector at a time. Actually, however, for performance and efficiency reasons, DOS reserves space for your files using groups of connected sectors called allocation units or clusters. Depending on your disk size, the number of sectors that make up an allocation unit will differ. Typically, most hard disks will use either four or eight sectors per allocation unit, as shown here:





An allocation unit is the smallest amount of disk space DOS can reserve for a file. Depending on your file size, DOS may need to reserve one allocation unit or several. In most cases, a portion of the allocation unit remains unused. Figure 25-1 illustrates how DOS would store three files of 1, 1024, and 2048 bytes, respectively. Note the amount of unused space in each allocation unit.

Figure 25-1 illustrates that no matter how small your file size appears in the directory listing, DOS must still allocate a cluster of disk space to store the file. If you create several very small files, much of the allocation unit is unused. In Chapter 42 you will better understand why DOS uses allocation units when you examine disk fragmentation. For now, keep in mind that creating several small files actually uses up more disk space than a large file containing the same information.

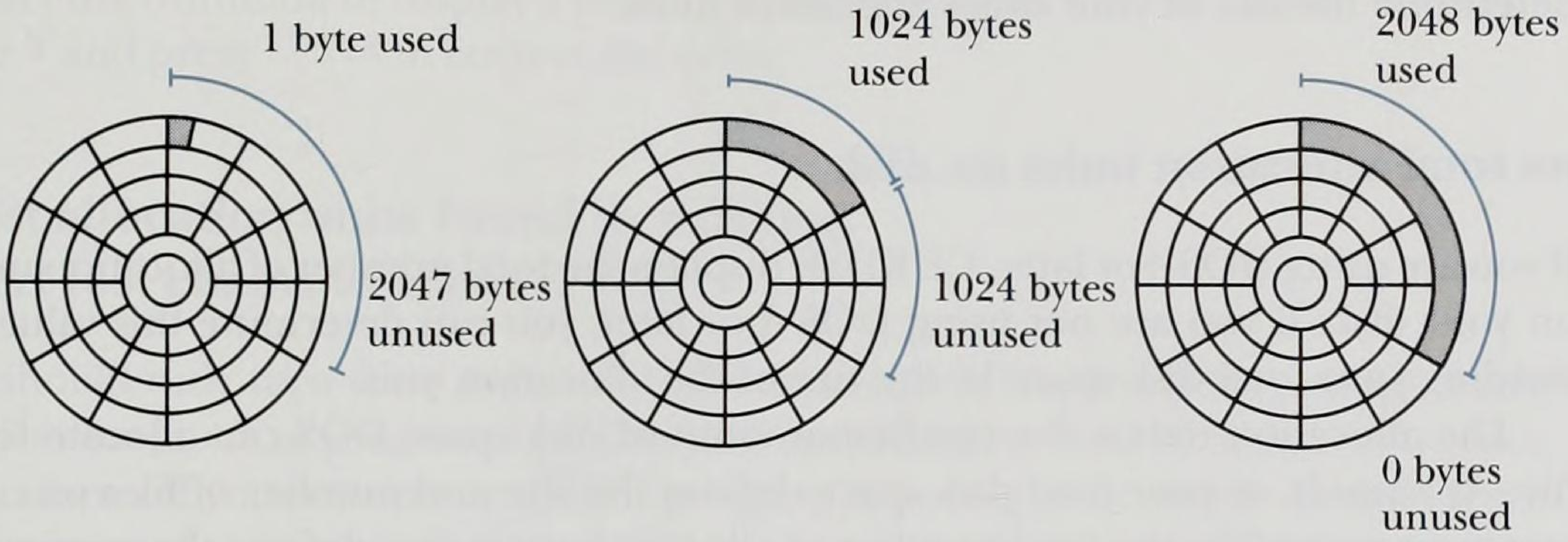
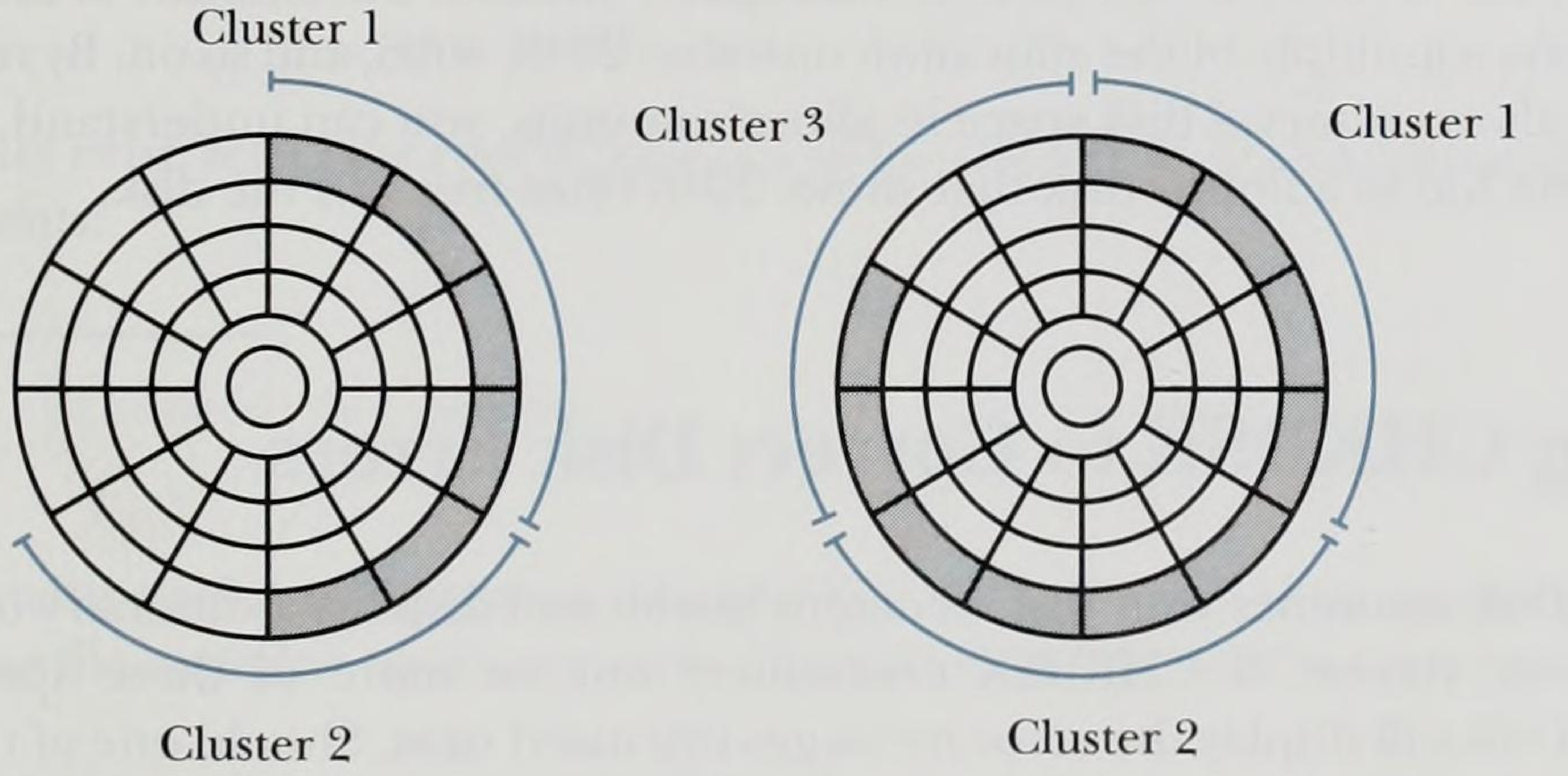

1 byte used
2047 bytes unused
1024 bytes used
1024 bytes unused
2048 bytes used
0 bytes unused

Figure 25-1. *Most files only use a portion of the allocation unit*

If your file size exceeds the size of an allocation unit, DOS reserves the number of allocation units required. Figure 25-2 illustrates how DOS might store two larger files.

If you are using DOS 4 or later, CHKDSK displays the disk's cluster size. If you are not using DOS 4, invoke either CHKDSK or DIR and note the amount of available disk space. Next, create a small file containing only a byte or two. Use DIR or CHKDSK once again to note the amount of available disk space. Rather than consuming the number of bytes listed in the file's directory listing, DOS has reserved the number of bytes in an allocation unit. By subtracting the current number of bytes available


Cluster 1
Cluster 2
Cluster 3
Cluster 1
Cluster 2

A 3072-byte file requires two clusters

A 5120-byte file requires three clusters

Figure 25-2. *DOS reserves the number of allocation units necessary*

from the number of bytes that were available before you created the file, you can determine the size of your disk's allocation units.

nnnn total allocation units on disk

If you are using DOS 4 or later, CHKDSK displays the total number of allocation units on your disk. If you are not using DOS 4 or later, you can determine this value by dividing your total disk space by the size of the allocation unit.

The allocation unit is the smallest amount of disk space DOS can allocate for a file. As a result, as your total disk space defines the size and number of files you can create on your disk, the total number of allocation units also defines the maximum number of files. For example, assume you have a 20Mb hard disk, with 10,405 allocation units. If you wanted to store only files containing a single byte, your disk size would lead you to believe your disk can store 20 million 1-byte files. However, because each file requires a minimum of one allocation unit, the maximum number of 1-byte files the 20Mb disk can store is 10,405, which happens to match the total number of disk allocation units.

nnnn available allocation units on disk

Again, in DOS 4 or later, CHKDSK displays the number of available allocation units. If you are not using DOS 4 or later, you can determine this value by dividing your available disk space by your disk's allocation unit size.

As you work with DOS files, pay close attention to the amount of available space on your disk. Assume for example, you copy files to a floppy disk until the disk fills. If you perform DIR commands between each file copy, you will find the disk never has 5, 10, or 50 bytes of available disk space. Instead, the amount of available space is always a multiple of the allocation unit size: 2048, 4096, and so on. By remembering DOS always reserves disk space in allocation units, you can understand why copying a 5-byte file to a floppy disk that shows 2048 bytes free fills the disk.

Using CHKDSK to Correct Disk Errors

CHKDSK examines your disk's current health and displays a summary of its findings on your screen. If CHKDSK encounters one or more of three specific errors, CHKDSK will display the error messages discussed next. Should one of the following error messages occur, you must invoke CHKDSK a second time, using the /F switch:

```
C:\> CHKDSK /F <ENTER>
```

After CHKDSK displays the error messages, CHKDSK will either automatically correct the condition or display a prompt asking you if you want to correct the error. Type Y and press ENTER to correct the error.

nnn lost allocation units found in *n* chains. Convert lost chains to files (Y/N)?

You should only turn your computer off when DOS is displaying its prompt. Many applications, such as your word processor or spreadsheet, store information on disk. If you turn off your computer while a program is storing information to disk or has files open, only a portion of the information may actually get recorded, leaving your disk in an inconsistent state. A *lost cluster* (allocation unit) is information stored on your disk that does not belong to a file. Lost clusters typically occur when you turn off your computer while a program is running.

When it encounters lost clusters, CHKDSK gives you a chance to store the clusters in files you can later edit using your word processor or the DOS editor. In some cases, the files may contain useful information. In most cases, however, you will simply delete the files using DEL to free up the disk space the lost clusters consumed.

When you convert lost clusters to files, CHKDSK creates the files in your root directory using the names FILE0000.CHK, FILE0001.CHK, and so on. The number of files CHKDSK creates depends on the number of *chains* of lost clusters. If CHKDSK finds one chain, CHKDSK will create one file. If CHKDSK encounters two chains, CHKDSK will create two files, and so on.

NOTE

CHKDSK only converts lost clusters to files when you use the /F switch.

After the files exist, select the root directory and use the TYPE command to display the file's contents:

```
C:\> TYPE  FILE0000.CHK <ENTER>
```

If the file contains useful information, rename the file and store it in the proper directory. If the file contains unrecognizable characters, delete the file with DEL.

REMEMBER

If you only turn off your computer from the DOS prompt, you greatly reduce the chance of lost clusters.

FILENAME.EXT allocation error, size adjusted

As you know, each file's directory entry tracks the file's size in bytes. By dividing the file's size by the size of your disk's allocation units, you can determine the number of disk allocation units the file requires. CHKDSK compares each file's size to the number of allocation units it finds for the file. In most cases, CHKDSK will find the number of allocation units is correct. However, if you have turned off your computer while a program is running or used the CTRL-C keyboard combination to end a program that writes information to disk, CHKDSK may find an inconsistency. When this occurs, CHKDSK changes the file's directory entry size to match the number of allocation units found.

This error message normally implies you have lost data. If you can display the file's contents using TYPE or your word processor, do so and determine the amount of information lost. If the file contains a command, do not execute it. As you'll recall, a program is a list of instructions your computer executes. If the program now contains too few or too many instructions, running the program may destroy other information on disk. Copy the command from your original floppy or restore the file from your disk backups, as discussed in Chapter 22.

FILENAME.EXT is cross linked on allocation unit *nn*

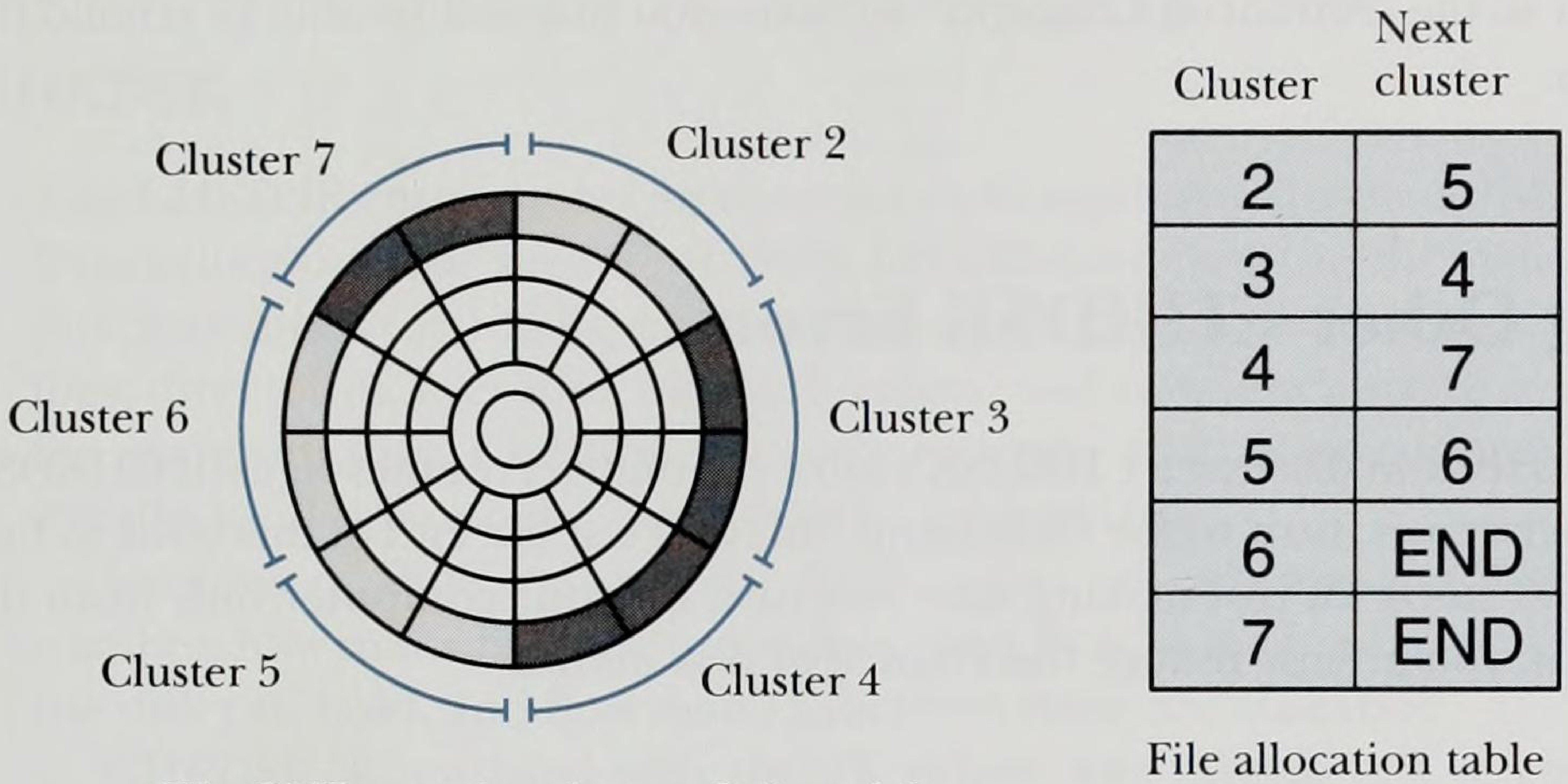
DOS stores files on disk using chains of clusters (allocation units). In Chapter 33 you will learn DOS uses a special table called the file allocation table or FAT to locate each file's clusters. As it examines your disk, CHKDSK makes sure two files aren't trying to use the same cluster. When two or more files think they own the same cluster, the files are *cross-linked.* As before, cross-linked files typically occur when you turn off your computer when a program is running or when you end a program that writes to disk by using the CTRL-C key combination.

Figure 25-3 illustrates two files, each containing three clusters. The figure includes a simple file allocation table that shows each file's clusters. For simplicity, the figure uses two-sector clusters.

Assume you were editing file ONE and you end the editing session using CTRL-C As a result, DOS incorrectly updates the file allocation table, cross-linking the files on cluster 3, as shown in Figure 25-4

In this case, file ONE thinks it owns clusters 2, 3, 4, and 7. Likewise, file TWO thinks it owns clusters 3, 4, and 7. Clusters 5 and 6 have become lost clusters, not belonging to any file. If you edit file ONE, the file will contain part of its original contents and part of file TWO's contents. Worse yet, if you delete file ONE, you will also delete part of file TWO. As you can see, cross-linked files are a nightmare.

The easiest way to correct cross-linked files is to invoke CHKDSK with the /F switch to first recover the lost clusters. Next, if you have backup copies of both files,

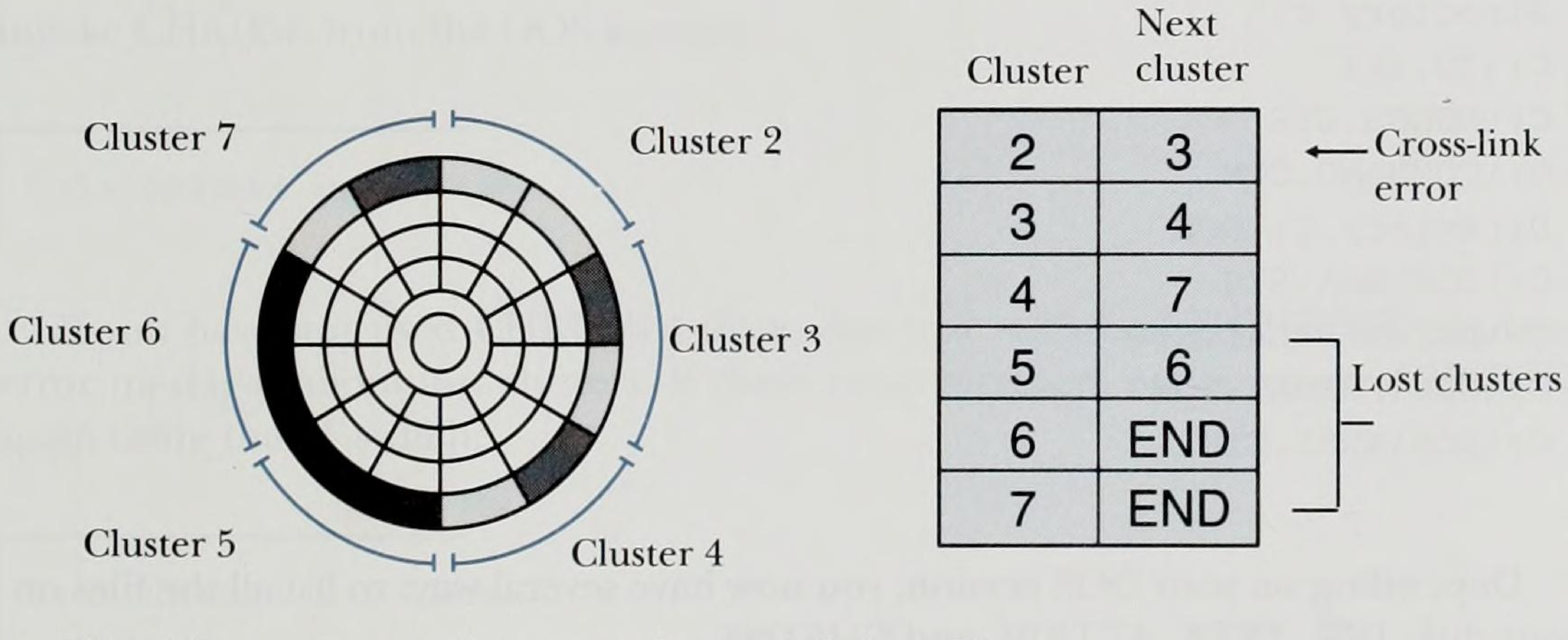

Cluster 7
Cluster 2
Cluster 6
Cluster 3
Cluster 5
Cluster 4

Cluster	Next cluster
2	5
3	4
4	7
5	6
6	END
7	END

File allocation table

File ONE contains clusters 2, 5, and 6
File TWO contains clusters 3, 4, and 7

Figure 25-3. *Two files, each containing three clusters*


Cluster 7
Cluster 2
Cluster 6
Cluster 3
Cluster 5
Cluster 4
Cross-link error
Lost clusters

Cluster	Next cluster
2	3
3	4
4	7
5	6
6	END
7	END

File ONE, due to cross-link, owns clusters 2, 3, 4, and 7
File TWO originally owned clusters 3, 4, and 7

Figure 25-4. *Files cross-linked on cluster 3*

delete the current copies and restore the current backups. If you don't have backup copies, turn to the "Advanced Concepts" section—you may still be able to rebuild the original files.

Handling Other CHKDSK Errors

The previous section discussed CHKDSK's most common error messages. If CHKDSK displays other errors, turn to the Command Reference at the end of this book to find the corrective steps. By just making sure you turn off your computer only from the DOS prompt, you greatly reduce the chance of disk errors.

Using CHKDSK to Display Filenames

To perform its processing, CHKDSK must read every directory on your disk. By invoking CHKDSK with the /V switch, you can direct CHKDSK to display the name of every file on your disk, as shown here:

```
C:\> CHKDSK  /V <ENTER>

Volume DOS         created 03-29-1993 10:08a
Volume Serial Number is 1661-AB75
Directory C:\
C:\IO.SYS
C:\MSDOS.SYS
C:\COMMAND.COM
Directory C:\DOS
C:\DOS\EGA.SYS
C:\DOS\DISPLAY.SYS
C:\DOS\FORMAT.COM
C:\DOS\ANSI.SYS
```

Depending on your DOS version, you now have several ways to list all the files on your disk: DIR, TREE, ATTRIB, and CHKDSK.

SUMMARY

CHKDSK

The CHKDSK command is an external command that checks a disk's status. Depending on your version of DOS, CHKDSK will tell you the name of the disk; how much total disk space there is; how much disk space is used in hidden files, directories, user files, and bad sectors; and how much space is available for any given disk. CHKDSK also tells you how much conventional RAM is installed and how much is available for program use.

Versions of DOS starting with DOS 4 also tell you about a disk's serial number, how many bytes per allocation unit, how many total allocation units the disk can hold, and how many allocation units are available.

CHKDSK, when used with the /F switch, can inform you of and often retrieve lost clusters, which can occur when a program that writes files to disk terminates before it can finish writing all of the file it was processing. It's best to have good backups, rather than depending on retrieved files, however.

CHKDSK can also be used as another way to list filenames, by using the /V switch.

Hands On

Invoke CHKDSK from the DOS prompt.

```
C:\> CHKDSK <ENTER>
```

If you have not used CHKDSK before, don't be surprised if CHKDSK displays error messages about lost clusters. If these error messages occur, invoke CHKDSK again using the /F switch.

```
C:\> CHKDSK  /F <ENTER>
```

Type **Y** when CHKDSK asks you if it should convert the lost cluster chains to files.

If you are using DOS 4 or later, note the size of your disk's allocation unit. Remember, every time you create a small file on your disk, CHKDSK reserves the number of bytes required for an allocation unit, not the actual file size.

Last, invoke CHKDSK using the /V switch to display the names of all the files on your disk.

```
C:\> CHKDSK  /V <ENTER>
```

Review

1. What is an allocation unit?
2. When should you invoke CHKDSK?
3. What is the purpose of CHKDSK's /F switch?
4. What is a lost cluster? How do you correct lost clusters?
5. What is a cross-linked file? How do you correct cross-linked files?

Answers

1. An allocation unit is a collection of consecutive sectors. An allocation unit (sometimes called a disk cluster) defines the smallest amount of disk space DOS can allocate for a file.
2. You should invoke CHKDSK on a regular basis to examine your disk's current health. In fact, many users invoke CHKDSK from their AUTOEXEC.BAT to ensure CHKDSK executes on a daily basis.
3. If CHKDSK encounters errors it can correct, the /F switch directs CHKDSK to fix the errors. If you don't include the /F switch in the CHKDSK command line, CHKDSK will not correct the disk errors, regardless of how you respond to the various CHKDSK prompts.
4. DOS stores a file's information in one or more allocation units or clusters. A lost cluster is a cluster of information stored on your disk that does not belong to a file. Lost clusters typically occur when you turn off your computer while a program is running. If CHKDSK encounters lost clusters, CHKDSK will display a message showing the total number of allocation units in lost chains and the number of lost chains. Invoke CHKDSK with the /F switch to convert

the lost clusters into files. CHKDSK will store each cluster chain in a root directory file with the name FILE*nnnn*.CHK (for example, FILE0000.CHK and FILE0001.CHK). Use TYPE or your word processor to display each file's contents. If the file contains useful information, rename the file and store the file in the proper directory. If the file contains unrecognizable characters, delete the file to free up the disk space the lost clusters consumed.

5. Like lost clusters, cross-linked files may occur when you turn off your computer while a program is running or use the CTRL-BREAK keyboard combination to end a program that is writing to disk. The best way to correct two cross-linked files is to delete the files and restore your backup copies of the files. If backup copies are unavailable, and both cross-linked files contain text that you can easily distinguish, you may be able to rebuild the original files as discussed in the "Advanced Concepts" section.

Advanced Concepts

Chapter 33 examines in detail how DOS stores files on your disk. As you read through Chapter 33, consider the different steps CHKDSK actually performs, such as searching the file allocation table for clusters marked invalid by FORMAT, for lost clusters that don't belong to a file, and for cross-linked files. To perform this process CHKDSK must read each file's directory entry, locate the file's corresponding clusters in the file allocation table, and then compare the two for consistency. Considering your disk may contain a very large number of files, CHKDSK performs a lot of work in a very short time period.

How DOS Knows the Allocation Unit Size

Depending on your disk type, the size of your disk's allocation unit will differ. The question you should ask yourself is how DOS knows the correct allocation unit size. As discussed in Chapter 7, every time you format a disk, FORMAT uses the disk's first sector as a boot record. If you format the disk as bootable, the boot record contains a small program that begins the DOS startup process. If the disk is not bootable, the boot record contains a small program responsible for the following error message:

```
Non-System disk or disk error
Replace and strike any key when ready
```

In addition to these programs, your disk's boot record contains information about the disk, such as the sector size, total number of sectors, and the size of each disk allocation unit, as shown in Figure 25-5. Using the boot record, DOS knows the disk's specifics.

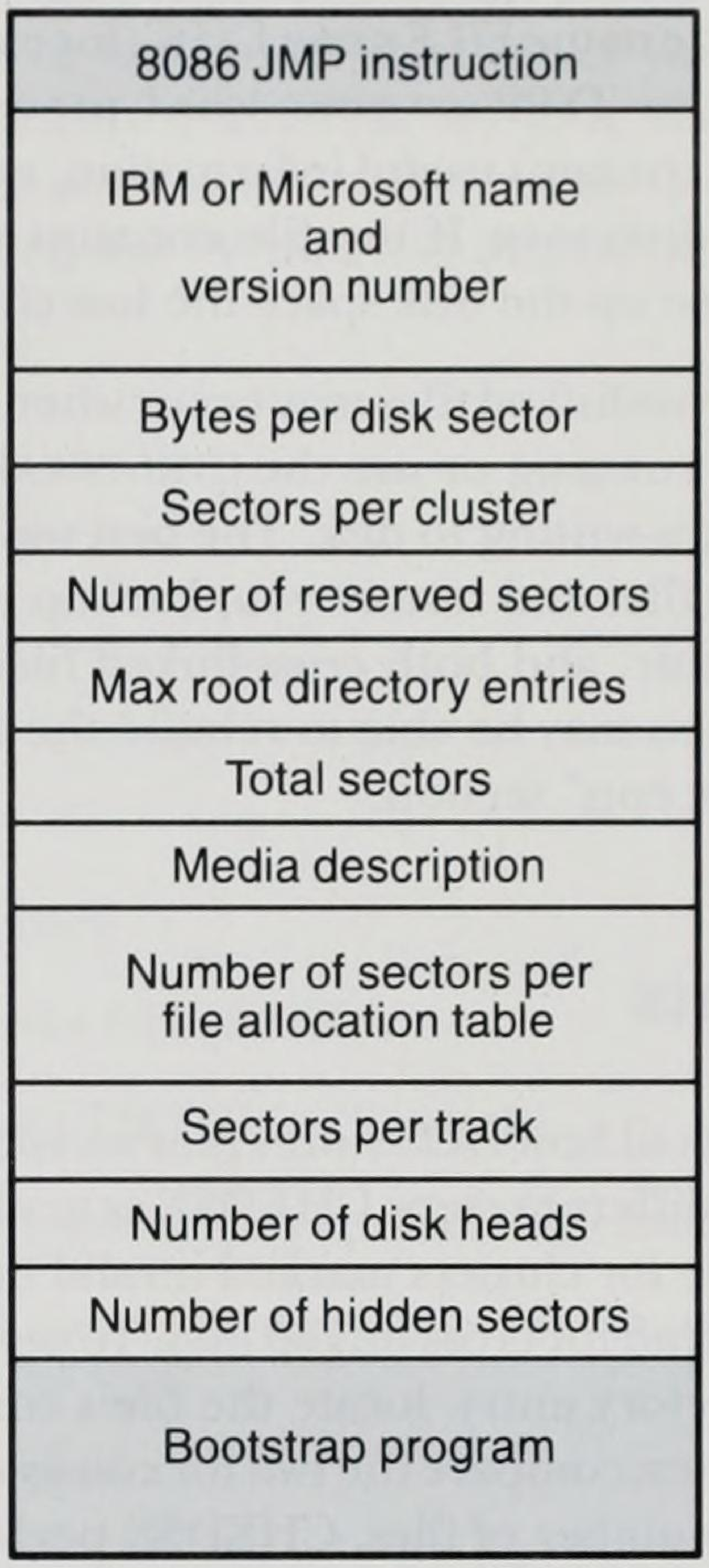

8086 JMP instruction
IBM or Microsoft name and version number
Bytes per disk sector
Sectors per cluster
Number of reserved sectors
Max root directory entries
Total sectors
Media description
Number of sectors per file allocation table
Sectors per track
Number of disk heads
Number of hidden sectors
Bootstrap program

Figure 25-5. *Entries in the DOS boot record*

Rebuilding Cross-Linked Files

Two files become cross-linked when one of the files thinks it owns one or more clusters actually owned by the other. The most common cause of cross-linked files is turning off your computer in the middle of a command or using CTRL-C to end a command that is writing to disk. Figure 25-4 illustrates two cross-linked files. The best way to correct cross-linked files is to delete the files and restore backup copies. If backup copies aren't available, and the files contain only text, you can rebuild the files by following the steps presented here.

First, when two files become cross-linked, you normally also have lost clusters. Using the /F switch, invoke CHKDSK to recover the lost clusters, as previously discussed.

Second, copy the currently cross-linked files to a new directory or floppy disk. Because DOS allocates unique clusters for the file copies, the files no longer share clusters. However, both of the new files will contain all the information in the cross-linked clusters. One file will be correct, one file will have the wrong clusters.

Next, delete the cross-linked files. Using the TYPE command or your word processor, determine which of the two new files contains the correct information and copy that file back to the current directory. Using your word processor or the DOS editor, edit the second file, removing the invalid information the file contains due to the previous cross-link.

Last, using your word processor, insert the previously lost clusters back into the file. The file is now correct, and you can copy its contents back to the current directory.

Delete the file FILE000*n*.CHK that contained the lost clusters, as well as the copies of the files you created in step 2. To rebuild files with cross-linked clusters, each file's contents must contain text you can easily distinguish. If you tried to edit two cross-linked command files, for example, each file would contain meaningless characters that prevent you from determining which information to delete and which information to add. For cross-linked command files, you have to rely on your backup copies.

Key Terms

Allocation unit The smallest amount of disk space DOS can allocate for a file. Allocation unit sizes are multiples of your disk sector size, such as 512, 1024, 2048, or 4096 bytes.

Cluster A disk allocation unit.

Lost clusters Disk clusters that contain information, but do not belong to a DOS file. Lost clusters are a result of a disk error.

Cross-linked files Two or more files that, due to a disk error, think they own the same cluster or clusters.

Chapter 26

DOS Pretender Commands

Throughout this book you have used your hard disk drive C, the floppy drive A, and possibly drive B if your computer has two floppy disks. Because these disk drives physically connect to your computer, they are physical drives. This chapter examines the DOS *pretender commands* ASSIGN, SUBST, and JOIN. The commands are "pretenders" because they trick DOS into thinking one disk drive is another, a directory is actually a disk drive, or a disk drive is actually a directory.

DOS provides pretender commands to help you run older programs that only know about drives A and B, or older programs that don't support DOS subdirectories. Pretender commands also let you define *logical disk drives.* Like physical drives, you access a logical drive using a drive letter and colon. Unlike physical drives you can see and touch, logical drives correspond to a DOS directory or, in the case of RAM drives, an electronic location. For example, the logical drive E may correspond to the directory \WP\LETTERS. Once you define the logical drive E, you can use the drive letter instead of typing the entire directory name.

Disk Drive Identity Swapping with ASSIGN

The original IBM PC came with the floppy disk drives A and B. As such, many older applications were written to only use drive A or B. When the IBM PC XT first made hard disks available, many of these programs would not run on the hard disk. To help users run these older programs on their hard disk, DOS introduced the ASSIGN command. ASSIGN is an external command that lets you trick DOS into thinking one disk drive is another.

DOS 6 does not support the ASSIGN command.

NOTE

Assuming a program only looks for its files on drive A, you can instruct DOS to direct all references for drive A to drive C with the following statement:

```
C:\> ASSIGN  A=C <ENTER>
```

After you issue this command, DOS automatically uses drive C when a command references drive A. If the older program tries to open a file on drive A, DOS automatically uses drive C instead. For example, assume you issue the following DIR command:

```
C:\> DIR  A: <ENTER>
```

Rather than displaying the files on the disk in drive A, DOS displays the files on drive C.

To remove the drive assignment, invoke ASSIGN without a command line.

```
C:\> ASSIGN <ENTER>
```

A directory of drive A will now list the files on drive A as before.

ASSIGN lets you perform multiple drive assignments at one time. The following command, for example, assigns both floppy drives A and B to drive C:

```
C:\> ASSIGN  A=C  B=C <ENTER>
```

As before, you remove the assignments by issuing any other ASSIGN command.

If you have a program that only looks to drive A for its files, your command sequence would be the following:

1. Use ASSIGN to assign drive A to drive C.
2. Run the program.
3. Use ASSIGN to remove the drive assignment.

If you are using DOS 5, you can use the /STATUS switch to display the current disk assignments as follows:

```
C:\> ASSIGN  /STATUS <ENTER>
Original A: set to C:
```

NOTE

Many DOS commands that perform low-level disk operations do not support logical disk drives and drive reassignments. Table 26-1 lists the commands you should not use with ASSIGN, SUBST, and JOIN. If you are using DOS 3.1 or later, Microsoft recommends that you use the SUBST command discussed next to perform your disk drive reassignments.

Substituting a Drive Letter for a Directory Name with SUBST

Just as many older software programs didn't support disk drives other than A or B, many older programs didn't support DOS subdirectories. If you tried to use a directory path with such a program, the program would display an error message such as "File not found" or "Invalid file name." To let you use directories with such programs, DOS 3.1 introduced the SUBST command. SUBST is an external DOS command that lets you substitute a logical disk drive letter for a directory path.

The following command substitutes the logical drive letter E for the DOS directory:

```
C:\> SUBST  E:  C:\DOS <ENTER>
```

BACKUP
CHKDSK
DISKCOMP
DISKCOPY
FDISK
FORMAT
LABEL
MIRROR
RECOVER
RESTORE
SYS
UNFORMAT
UNDELETE

Table 26-1. *Do Not Use These DOS Commands with ASSIGN, SUBST, or JOIN*

Once you define a logical drive, you can use its drive letter as you would any disk drive. The following command, then, lists the files on drive E:

```
C:\> DIR  E: <ENTER>
```

When DOS encounters the logical drive letter in this case, it automatically substitutes the corresponding directory path, and displays the files in the DOS directory:

```
C:\> DIR E: <ENTER>

 Volume in drive E is DOS
 Volume Serial Number is 1A54-45E0
 Directory of E:\

.            <DIR>     11-23-92   9:26p
..           <DIR>     11-23-92   9:26p
DBLSPACE BIN     50284 02-12-93   6:00a
FORMAT   COM     22717 02-12-93   6:00a
NLSFUNC  EXE      7036 02-12-93   6:00a
  :        :        :       :        :
QBASIC   INI       132 02-20-93  12:10p
MOUSE    INI        28 02-03-93  11:54a
DEFAULT  CAT        66 02-08-93   7:51p
      128 file(s)    5826105 bytes
                    16762880 bytes free
```

Drive E is called a logical disk drive because it does not physically exist inside your computer, only logically. Instead, DOS lets you use the logical drive, pretending such a disk drive exists.

If you invoke SUBST without a command line, SUBST will display the current drive substitutions.

```
C:\> SUBST <ENTER>
E: => C:\DOS
```

If you select drive E as the current drive, you select the substituted directory as the current directory.

To remove a substitution, invoke SUBST with the logical drive letter and the /D switch, as shown here:

```
C:\> SUBST  E:  /D <ENTER>
```

SUBST will not let you remove a drive substitution if the substituted drive is the current drive.

NOTE

By default, drive E is the last drive letter DOS lets you use. If you need to assign a drive letter greater than E, you must use the CONFIG.SYS LASTDRIVE entry discussed in Chapter 24.

Although the DOS developers intended for SUBST to help users run programs that don't support directories, users commonly use SUBST to abbreviate long directory names.

Abbreviating Long Directory Names with SUBST

As you create new directories to organize your files, your directory tree will grow. Assume, for example, you constantly switch between the directories \WP\LETTERS\BUSINESS and \123\EXPENSES\1STQTR. You can see that repeatedly typing the long directory path names could become quite frustrating. Using SUBST, however, you can first assign the directory \WP\LETTERS\BUSINESS to drive E, as shown here:

```
C:\> SUBST  E:  \WP\LETTERS\BUSINESS <ENTER>
```

Next, you can substitute drive F for the directory \123\EXPENSES\1STQTR:

```
C:\> SUBST  F:  \123\EXPENSES\1STQTR <ENTER>
```

With your directories labeled as drives E and F, you can quickly switch between the two directories.

```
C:\> E: <ENTER>
E:\> F: <ENTER>
F:\>
```

Although you should only include directories most likely to include your external commands within the command path, many users often push PATH's 122-character limit. By assigning logical disk drive letters to the longer PATH entries, you can reduce your current path length, letting you add entries. When DOS examines the command path, DOS automatically substitutes the correct directory path for each drive letter. Remember, however, the more directories you include in your command path, the more time DOS must spend searching the directories.

As before, do not use SUBST with the DOS commands listed in Table 26-1.

Making DOS Think a Drive Is a Directory with JOIN

Between the ASSIGN, SUBST, and JOIN commands, users typically have the most difficulty understanding JOIN. The JOIN command is essentially the opposite of SUBST. The JOIN command lets you trick DOS into thinking a disk drive is a directory on the current disk.

To understand why you would want to join a drive to your directory tree, consider the DOSSHELL program. The DOSSHELL provides a menu-driven interface to your directory structure and DOS commands. Each time you change the current drive, DOSSHELL reads all the files and directories on the selected disk and displays a directory tree. If you change drives on a regular basis, this constant reading of the directory structure not only becomes frustrating, but also consumes a lot of time. When you change directories, however, DOSSHELL doesn't read the directory tree.

DOS 6 does not support the JOIN command.

NOTE

Using the JOIN command, you can convince DOS that another commonly used drive is a directory, and can avoid the constant reading of the directory tree.

To use JOIN, you must first create an empty directory on your hard disk to which you will join the drive. In this case, you will join drive A to the directory A_DRIVE. Create the directory using MKDIR.

```
C:\> MKDIR  \A_DRIVE <ENTER>
```

Next, use the following JOIN command to trick DOS into thinking drive A is actually the directory A_DRIVE:

```
C:\> JOIN  A:  \A_DRIVE <ENTER>
```

Place a floppy disk that contains files in drive A. Next, issue the following DIR command:

```
C:\> DIR  \A_DRIVE <ENTER>
```

Using the directory A_DRIVE, DOS will display the files on the floppy disk in drive A. Using CHDIR, you can select A_DRIVE as the current directory:

```
C:\> CHDIR  \A_DRIVE <ENTER>
C:\A_DRIVE>
```

If you invoke the TREE command, it will include the directory A_DRIVE, as well as the files in the directory:

```
C:\> TREE  C:\  /F <ENTER>
```

As you can see, DOS treats the disk in drive A as if the disk is part of the directory tree on drive C.

If you invoke JOIN without a command line, JOIN will display the drives it has joined, as well as the corresponding directories.

```
C:\> JOIN <ENTER>
A: => C:\A_DRIVE
```

To remove the drive from the directory tree, invoke JOIN with the drive letter and /D switch.

```
C:\> JOIN  A:  /D <ENTER>
```

Although you might not use JOIN for commands you issue from the DOS prompt, JOIN might improve the performance of different shell programs. As before, do not use JOIN with the DOS commands listed in Table 26-1.

Hands On

Place a floppy disk containing files in drive A. Issue the following DIR command to display the disk's contents:

```
C:\> DIR  A: <ENTER>
```

SUMMARY

ASSIGN

The ASSIGN command lets you redirect DOS from one disk drive to another. Some older programs only look for files on drive A or B. Using ASSIGN, you can trick these programs into using your hard disk. The following command directs DOS to use drive C for all references to drive A:

```
C:\> ASSIGN  A=C <ENTER>
```

After you run the program, invoke the ASSIGN command without a command line to remove the drive assignments. DOS 6 does not support the ASSIGN command.

SUMMARY

SUBST

The SUBST command was originally introduced to help you run older programs that don't support directories. SUBST lets you substitute a logical drive letter for a directory path. Today, most users use SUBST to abbreviate long directory names. The following command, for example, abbreviates a directory path with the logical drive letter E:

```
C:\> SUBST  E:  \WP\BUSINESS\MEMOS <ENTER>
```

After you define the logical drive, you can use the drive letter in your commands as you would any disk drive. To create logical drives beyond the default drive letter E, such as drives F, G, and H, you must include a LASTDRIVE entry in your CONFIG.SYS file, as discussed in Chapter 24.

JOIN

The JOIN command lets you trick DOS into thinking another disk drive is actually a directory in your directory tree. Many shell programs perform extra processing when you change disk drives that the programs don't perform when you change directories. By using JOIN to convince DOS another drive is actually a directory, you can improve the shell's performance. You must join a disk drive to an empty directory. The following command, for example, joins drive A to the empty directory A_DRIVE.

```
C:\> JOIN  A:  \A_DRIVE <ENTER>
```

After you join a disk to a directory, you can access the files the disk contains using standard directory operations. DOS 6 does not support the JOIN command.

Next, issue the following ASSIGN command directing DOS to route all references from drive A to drive C and then reissue the DIR command. If you are using DOS 6, which does not support the ASSIGN command, continue your reading with SUBST.

```
C:\> ASSIGN  A=C <ENTER>
C:\> DIR  A: <ENTER>
```

In this case, rather than displaying the files on drive A, DOS displays the files on drive C. If you are using DOS 5, invoke ASSIGN with the /STATUS switch.

```
C:\> ASSIGN  /STATUS <ENTER>
```

Remove the drive assignment by invoking ASSIGN without a command line.

```
C:\> ASSIGN <ENTER>
```

A directory listing of drive A displays, once again, the files on the floppy disk.

Next, issue the following SUBST command to substitute the logical drive E for the DOS directory:

```
C:\> SUBST  E:  \DOS <ENTER>
```

Use the following DIR command to display a directory listing for drive E:

```
C:\> DIR  E: <ENTER>
```

As you will see, DOS automatically substitutes the path C:\DOS for drive E, displaying your DOS files. Select drive E as the current drive and repeat the directory command.

```
C:\> E: <ENTER>
E:\> DIR <ENTER>
```

As you can see, DOS lets you use logical drives as you would any disk drive. Select drive C as the default and invoke SUBST to display the current substitutions.

```
E:\> C: <ENTER>
C:\> SUBST <ENTER>
```

Use the SUBST /D switch to remove the drive substitution.

```
C:\> SUBST  E:  /D <ENTER>
```

Next, if you are not using DOS 6 (which does not support JOIN) create the directory A_DRIVE using MKDIR.

```
C:\> MKDIR  \A_DRIVE <ENTER>
```

Use the following JOIN command to attach the disk that is in drive A to the current directory tree:

```
C:\> JOIN  A:  \A_DRIVE <ENTER>
```

Use the following DIR command to display the files on the disk in drive A:

```
C:\> DIR  \A_DRIVE <ENTER>
```

Use CHDIR to select A_DRIVE as the current directory, and issue a DIR command.

```
C:\> CHDIR  \A_DRIVE <ENTER>
C:\A_DRIVE> DIR <ENTER>
```

Select the root directory as the current directory and issue the following TREE command:

```
C:\A_DRIVE> CHDIR  \ <ENTER>
C:\> TREE  /F <ENTER>
```

As you can see, once you join a disk to a directory, you can access the disk's files using standard directory commands. Invoke JOIN without a command line to display the current joins.

```
C:\> JOIN <ENTER>
```

Invoke JOIN using the /D switch to remove the directory join.

```
C:\> JOIN  A:  /D <ENTER>
```

Last, using RMDIR, remove the directory A_DRIVE as shown here:

```
C:\> RMDIR  \A_DRIVE <ENTER>
```

Review

1. What is a logical disk drive?
2. What is a pretender command?

3. When would you use the ASSIGN command?
4. When would you use the SUBST command?
5. When would you use the JOIN command?

Answers

1. A logical disk drive is a drive that does not physically exist on your system, but that you can reference in your DOS commands. The SUBST command creates logical drives that correspond to existing directories. When you reference the logical drive letter, DOS automatically substitutes the corresponding directory name.
2. A pretender command is a command that tricks DOS into thinking that one disk drive is another, a drive letter is actually a subdirectory, or a disk drive actually resides as a directory in the current directory tree. DOS provides three pretender commands: ASSIGN, SUBST, and JOIN.
3. Some very old software programs only let you use floppy disk drives A and B. When you try to run these programs from your hard disk, the programs still look for their data files on the floppy drive. The ASSIGN command lets you trick DOS into using the hard disk instead of the floppy drives. DOS 6 does not support the ASSIGN command.
4. Some older software programs do not support directory path names, but do let you include a drive letter before a filename (such as E:FILENAME.EXT). Using the SUBST command, you can trick the program into using the directory. When DOS encounters the logical disk drive, DOS will automatically substitute the correct directory path name. Most users, however, use SUBST to abbreviate long directory names.
5. Many menu-driven shell programs display a directory tree for the current disk, letting you select files or directories from the tree using your keyboard or mouse. When you use the shell program to change default drives, the shell program will read the new drive's directory tree. Depending on the number of files on your disks and the number of times you change drives, this constant reading and rereading of directory trees can become quite tiresome. When you use these shell programs to change directories, however, the shell pro-

grams do not reread the directory tree. Using the JOIN command, you can trick DOS into thinking a disk drive is actually a directory in the current directory tree. In this way, you can reduce much of the extra processing shell programs perform. DOS 6 does not support the JOIN command.

Advanced Concepts

As you have learned, the SUBST command lets you create logical disk drives. If your computer only has a single floppy drive, DOS automatically creates a logical drive B to help you perform dual floppy disk commands.

By default, DOS assigns the drive letter A to your single floppy. However, if you use the drive letter B, in a DIR command for instance, DOS will prompt you to remove the current floppy disk and to insert the floppy disk you want to use in drive B, as shown here:

```
Insert diskette for drive B:
and press any key when ready
```

DOS now considers your floppy disk as drive B until you explicitly reference drive A once again. At that time, DOS will prompt you to insert the disk you want to use for drive A.

```
Insert diskette for drive A:
and press any key when ready
```

Using the logical drive B, you may be able to successfully run an older program that wants to use both drives, although your computer only has a single floppy.

Key Terms

Logical disk drive A disk drive that does not physically exist in your computer but that you define using the SUBST command. Logical disk drives help you abbreviate path names.

Pretender command The name given to the ASSIGN, SUBST, and JOIN commands because they make DOS pretend that one disk is another, a drive letter is a directory name, or a drive exists as a directory on the current disk. ASSIGN and JOIN are not supported by DOS 6.

Advanced Concepts

[illegible]

```
[illegible]
```

[illegible]

```
[illegible]
```

[illegible]

Key Terms

[illegible]

Chapter 27

DOS 5 and 6 DOSKEY and Macros

The discussion throughout the book to this point has related to all versions of DOS. For some commands, different DOS versions provide additional switches that improve the command's capabilities. This chapter and Chapter 28 present commands that are only available in DOS 5 or later. If you are not using DOS 5 or later, take a look at these chapters to get a feel for the advances DOS is making. This chapter presents the DOSKEY command, which remembers each command you enter, so you can quickly repeat them. In addition, DOSKEY lets you define single-line *macros,* which behave like one-line batch files, letting you abbreviate commonly used commands. Unlike DOS batch files stored on disk, DOSKEY macros reside in your computer's fast electronic memory, which means they execute very quickly.

Using DOSKEY to Recall Commands

In Chapter 5 you learned how to quickly repeat the previous command by pressing the F3 function key. As you learned, each time you execute a command, DOS stores the command in a buffer you can access with the function keys F1 through F4. Although this *previous-command buffer* is convenient, it only lets DOS keep track of the one previous command. The DOSKEY command lets DOS track many commands. The actual number of commands DOSKEY can track depends on the amount of memory you allocate to DOSKEY and the length (in characters) of the commands you execute.

DOSKEY is an external DOS command. The first time you execute the DOSKEY command, DOSKEY loads memory-resident software into your computer's memory. This software remains present in memory until you turn off your computer or restart DOS. When invoked from the DOS prompt, DOSKEY will display the following message, informing you that it has loaded its memory-resident software:

```
C:\> DOSKEY <ENTER>
DOSKEY installed.
```

By default, DOSKEY allocates 512 bytes of memory for a command buffer. Assuming the average length of your DOS commands is ten characters, DOSKEY will let you track approximately the previous 50 DOS commands. Should your DOSKEY buffer fill, DOSKEY will discard the oldest command in the buffer to make room for the new command.

Invoke the DATE, TIME, VER, and CLS commands. Using the UP ARROW and DOWN ARROW keys, you can cycle through your list of previous commands. UP ARROW directs DOSKEY to display the command you invoked immediately before the command now displayed. Likewise, DOWN ARROW directs DOSKEY to display the command you invoked immediately after the command being displayed. When no more commands precede or follow the command now displayed, DOSKEY ignores the arrow keys.

In addition to cycling through your previous command list with UP ARROW and DOWN ARROW, DOSKEY lets you use the PGUP and PGDN keys. PGUP directs DOSKEY to display the oldest command in the previous-command buffer, and PGDN directs DOSKEY to display your most recently used command. Table 27-1 summarizes how DOSKEY uses these keys to traverse the previous-command buffer.

Using the arrow keys described in Table 27-1, you can cycle through your list of previous commands. When you only need to go back a few commands, using the arrows is very convenient.

As the number of commands in DOSKEY's previous-command buffer becomes large, locating the desired command with the arrow keys can consume more and more time. Fortunately, DOSKEY provides several faster ways to locate a command.

Key	Function
UP ARROW	Displays the command invoked before command now displayed
DOWN ARROW	Displays the command invoked next after displayed command
PGUP	Displays the oldest command name in the command buffer
PGDN	Displays the most recently used command name

Table 27-1. *The DOSKEY Traverse Keys*

DOSKEY refers to the previous-command buffer as your *command history.* If you invoke DOSKEY using the /HISTORY switch, DOSKEY will display all the command names your previous-command buffer contains, as shown here:

```
C:\> DOSKEY /HISTORY <ENTER>
DATE
TIME
VER
CLS
DOSKEY /HISTORY
```

To reduce your typing, DOSKEY lets you abbreviate the /HISTORY switch as simply /H.

```
C:\> DOSKEY /H <ENTER>
```

If your previous-command buffer is large, DOSKEY lets you refer to commands by number. To begin, press F7, and DOSKEY will display your command history, preceding each command with its corresponding number.

```
1: DATE
2: TIME
3: VER
4: CLS
5: DOSKEY /HISTORY
6: DOSKEY /H
```

To select a command by number, press F9. DOSKEY will prompt you to type in the desired number as follows:

```
C:\> Line Number:
```

Type in the desired line number and press ENTER. In this case, if you type 4 and press ENTER, DOSKEY will display the CLS command.

```
C:\> CLS
```

DOS won't execute the command until you press ENTER. If you don't want to execute the command, press ESC. DOSKEY will erase the current command from your screen. As the number of characters in your commands becomes large, recalling commands in this way can be very convenient.

DOSKEY gives you one last way to recall a command. Simply type one or more letters of the command name and press F8. DOSKEY will then search the previous-command buffer, beginning with the letter or letters specified. For example, if you type V and press F8, DOSKEY will recall the VER command.

```
C:\> VER
```

As the number of commands in the previous-command buffer grows, two or more commands may begin with the letters specified. If you repeatedly press F8, DOSKEY will cycle through each of the matching commands.

If the number of commands in your previous-command buffer becomes unmanageably large, you can press ALT-F7 to discard the buffer's current contents: Once you empty the buffer in this way, the next command you invoke will become command number 1. Table 27-2 describes the DOSKEY function keys.

Function Key	Function
F7	Displays all command names in the buffer, preceding each with its corresponding number
F8	Displays the first command name that matches the letters you typed immediately before pressing F8. If several command names match, repeatedly pressing F8 will cycle through the matching names
F9	Prompts you to type in a command's number, as displayed by the F7 function key. If the number does not exist, displays the most recent command
ESC	Removes the current command name, leaving only the DOS prompt
ALT-F7	Discards the current buffer contents

Table 27-2. The DOSKEY Function Keys

Editing a Previous Command

Once you recall the desired command, you can execute the command by pressing ENTER, or you can edit the command as you did in Chapter 5. When you recall a command, DOSKEY places the cursor immediately to the right of the command, so you can easily execute the command by pressing ENTER. To help you edit the command, DOSKEY lets you use several editing keys. RIGHT ARROW and LEFT ARROW let you move the cursor one character position to the right or the left. BACKSPACE backs over characters, rubbing them out. To delete characters, you can also move the cursor to the position of the character you want to delete and press DEL. To insert characters into the command, place the cursor in the position at which you want to insert the text. Press INS until DOSKEY displays a larger cursor indicating insert mode. Type in the desired characters.

In addition to RIGHT ARROW and LEFT ARROW, DOSKEY lets you move through the command a word at a time by holding down the CTRL key and pressing either RIGHT ARROW or LEFT ARROW The CTRL-RIGHT ARROW key combination moves the cursor one word to the right, whereas CTRL-LEFT ARROW moves the cursor one word to the left. In addition, HOME moves the cursor to the command's first letter, and END moves the cursor to the end of the command. Table 27-3 summarizes DOSKEY's editing keys.

Other DOSKEY Switches

In addition to the /HISTORY switch, which lets you display the current contents of the previous-command buffer, DOSKEY supports four other switches: /BUFSIZE, /REINSTALL, /INSERT, and /MACROS.

Key	Function
RIGHT ARROW	Moves the cursor one character position to the right
LEFT ARROW	Moves the cursor one character position to the left
CTRL-RIGHT ARROW	Moves the cursor one word to the right
CTRL-LEFT ARROW	Moves the cursor one word to the left
INS	Toggles insert mode on and off
DEL	Deletes the character at the current user position
HOME	Moves the cursor to the start of the command
END	Moves the cursor to the end of the command

Table 27-3. The DOSKEY Editing Keys

DOSKEY's /BUFSIZE switch lets you specify, in bytes, the amount of memory you want DOSKEY to allocate for the previous-command buffer. By default, DOSKEY allocates 512 bytes. The larger you make DOSKEY's buffer, the more commands the buffer can store. For most users, a 512-byte buffer should be enough. If you make the buffer size too large, you may waste memory DOS could put to better use in another way. The minimum buffer size is 256 bytes. The following DOSKEY command selects a large buffer size of 1024 bytes:

```
C:\> DOSKEY /BUFSIZE=1024 <ENTER>
```

Because DOSKEY loads memory-resident software and allocates the buffer the first time you invoke it, you must include the /BUFSIZE switch when you first invoke DOSKEY or you must use the /REINSTALL switch discussed next.

In Chapter 14 you learned that to change PRINT's memory-resident settings, you needed to restart DOS. If you decide you want to change DOSKEY's buffer size for example, you can restart DOS and invoke DOSKEY with the new buffer size, or you can specify the new buffer size and include the /REINSTALL switch as shown here:

```
C:\> DOSKEY /REINSTALL /BUFSIZE=640 <ENTER>
DOSKEY installed.
```

The /REINSTALL switch directs DOSKEY to unload its current memory-resident software and reload itself using the new switch values. When you reinstall DOSKEY in this way, you discard the current contents of the previous-command buffer.

By default, when you edit a previous command you must first press the INS key to select insert mode. If you don't press INS, you will be in overtype mode, and will overwrite letters to the right of the cursor as you type. Depending on your preference, you can direct DOSKEY to automatically begin in insert mode so you don't have to press INS. To do so, invoke DOSKEY with the /INSERT switch.

```
C:\> DOSKEY /INSERT <ENTER>
```

As was the case with /BUFSIZE, you can only specify /INSERT the first time you invoke DOSKEY or you must again use /REINSTALL.

The "Advanced Concepts" section discusses how to create DOSKEY macros, which are similar to single-line batch files. At that time, you will use DOSKEY's /MACROS switch to display your defined macros.

SUMMARY

DOSKEY

The DOSKEY command lets you quickly recall your previous commands. DOSKEY provides a buffer in your computer's fast electronic memory that stores a list of your previously entered commands. Depending on the amount of memory you allocate to DOSKEY and the number of characters in each command, the actual number of commands DOSKEY can track will differ. Using your keyboard's UP ARROW and DOWN ARROW keys, you can quickly cycle through the list of commands.

Hands On

If you have not already done so, invoke DOSKEY.

```
C:\> DOSKEY <ENTER>
```

Next, invoke the DIR, TIME, CLS, and DATE commands. Using the UP ARROW and DOWN ARROW keys, cycle through your list of previous commands. Press PGUP to display the oldest command in the buffer. Use PGDN to display the most recently used command. Next, use DOSKEY's /HISTORY switch to display your command history.

```
C:\> DOSKEY  /HISTORY <ENTER>
```

Repeat the command using the /H abbreviation.

```
C:\> DOSKEY  /H  <ENTER>
```

Press the F7 function key to display the previous-command buffer, each command name preceded by its corresponding number. Select a command and use F9 to recall the command by number. Press ESC to remove the command name from the prompt.

Type **C** and press F8 to recall the CLS command. Type **D** and press F8. DOSKEY will highlight the DATE command. Press F8 again and DOSKEY will highlight DIR.

Last, press ALT-F7 to discard the current contents of the previous-command buffer.

Review

1. How many commands can the DOSKEY buffer store?
2. List three ways to recall a command with DOSKEY.
3. What happens when DOSKEY runs out of buffer space?
4. How does DOSKEY affect the behavior of the function keys F1 through F6?
5. What is a DOSKEY macro?

Answers

1. The number of commands DOSKEY can store depends on the amount of memory you allocate for DOSKEY's buffer, the number of characters in each command, and the number of DOSKEY macros you create. By default, DOSKEY uses a 512-byte buffer.
2. To recall a command with DOSKEY:
    a. Use the UP ARROW and DOWN ARROW to cycle through the previous-command buffer.

    b. Press F7 to show the number of each command in the previous-command buffer and then press F9, typing in the number of the desired command at DOSKEY's "Line Number:" prompt.

    c. Type the first few letters of the command name and then press F8.
3. DOSKEY will discard the oldest commands in the previous-command buffer to make space for new commands.
4. DOSKEY does not affect the behavior of the function keys F1 through F6. You can still use these function keys to edit the previous command that DOS buffers.
5. A DOSKEY macro is similar to a single-line batch file. Unlike batch files that reside on disk, DOSKEY macros reside in your computer's fast electronic memory, which means the macros execute faster than batch files.

Advanced Concepts

In addition to letting you quickly recall your previous commands, DOSKEY lets you define macros, which behave like single-line batch files. Like batch files, DOSKEY macros have a name. To execute the macro, you type the macro's name at the DOS prompt and press ENTER.

Unlike batch files that reside on disk, DOSKEY stores macros in your computer's fast electronic RAM. As a result, DOS can execute macros very quickly because DOS doesn't have to first read the macro from the slower mechanical disk. Unlike batch files that can contain many lines of commands, DOSKEY macros are restricted to a single line.

Because DOSKEY stores macros in RAM, the macros are lost each time you turn off your computer or restart DOS. If you create several macros you frequently use, place corresponding DOSKEY commands in your AUTOEXEC.BAT to define the macros each time your system starts.

The following DOSKEY command creates a macro named HIST that executes the DOSKEY /HISTORY command:

```
C:\> DOSKEY  HIST=DOSKEY  /HISTORY <ENTER>
```

After you define HIST, you can invoke the macro as follows:

```
C:\> HIST <ENTER>
```

DOS will then execute the DOSKEY /HISTORY command as desired. DOSKEY macros take priority over DOS internal commands, external commands, and batch files by the same name. For example, the following DOSKEY command creates a macro named CLS that invokes the ECHO command to display the message "Clear screen is redefined."

```
C:\> DOSKEY  CLS=ECHO Clear screen is redefined.
```

If you invoke CLS from the DOS prompt, your screen will display the following:

```
C:\> CLS <ENTER>
C:\> ECHO Clear screen is redefined.
Clear screen is redefined.
```

Because DOSKEY macros execute first, you may want to define macros for commands you don't want another user to access, such as FORMAT or DEL.

DOSKEY macros are not restricted to a single command, merely to a single line. For example, in Chapter 15 you created the following batch file, DATETIME.BAT, to clear your screen display and prompt you to enter the system date and time.

```
CLS
DATE
TIME
```

Because you can easily fit these three commands on one line, you can create a macro named DATETIME that issues all three commands as shown here:

```
C:\> DOSKEY  DATETIME=CLS  $T  DATE  $T  TIME <ENTER>
```

DOSKEY uses the $T symbol to separate multiple commands that appear on one line. To invoke the macro, type **DATETIME** at the DOS prompt, as shown here:

```
C:\> DATETIME <ENTER>
```

DOSKEY macros can also support command-line parameters and I/O redirection operators.

Accessing Command-Line Parameters

In Chapter 16 you used the batch file parameters %0 through %9 to access command-line information. In a similar way, DOSKEY macros use the symbols $1 through $9. The following macro SHOWEM uses the symbols $1 through $9 to display its command parameters.

```
C:\> DOSKEY  SHOWEM=ECHO  $1  $2  $3  $4  $5  $6  $7  $8  $9
```

If you invoke SHOWEM with the letters A, B, and C, your screen will display the following:

```
C:\> SHOWEM  A  B  C <ENTER>
C:\> ECHO A  B  C
A  B  C
```

As was the case with DOS batch files, if you invoke the macro with more than nine parameters, DOSKEY ignores the extras.

```
C:\> SHOWEM  A  B  C  D  E  F  G  H  I  J  K  L <ENTER>
C:\> ECHO  A  B  C  D  E  F  G  H  I
A  B  C  D  E  F  G  H  I
```

DOSKEY macros do not have a command like SHIFT that rotates parameters. Instead, if you want to access the entire command line, you must use the $* symbol, as shown here with the macro SHOWALL:

```
C:\> DOSKEY  SHOWALL=ECHO  $* <ENTER>
```

Using the $* symbol, the following macro, TYPE, combines the TYPE command with FOR to support multiple filenames and the DOS wildcards. Since a macro is higher in precedence than an internal command of the same name, it replaces the internal TYPE command.

```
C:\> DOSKEY  TYPE=FOR  %I  IN  ($*)  DO  TYPE  %I <ENTER>
```

Using the new TYPE, you can display the contents of several files in succession.

```
C:\> TYPE  CONFIG.SYS  AUTOEXEC.BAT <ENTER>
```

One DOSKEY macro cannot execute another. Therefore, when DOS encounters TYPE within the above FOR command, DOS correctly executes the internal TYPE command (not the macro), correctly displaying the file's contents.

Table 27-4 defines symbols you can use within DOSKEY macros.

Symbol	Meaning
$1 through $9	The macro's command-line parameters
$T	Separates multiple commands
$*	Represents the entire command line

Table 27-4. Symbols Supported by DOSKEY Macros

Using I/O Redirection Operators

As you have seen, the commands your macros execute can be very complex as long as they all fit on one line. As you create more powerful macros, you may eventually want to include I/O redirection operators within the macro command. Using the I/O redirection operators, however, is more difficult than you might think. The goal of the following macro, PRINTDIR, is to redirect the output of the DIR command to the printer.

```
C:\> DOSKEY  PRINTDIR=DIR  >  PRN <ENTER>
```

Unfortunately, when you execute this DOSKEY command, DOS assumes you want to redirect the output of DOSKEY to the printer, not the output of DIR.

If you want to use the I/O redirection operators within your DOSKEY macros, you must use the symbols $G (for greater-than symbol) for output redirection, $L (less-than) for input redirection, GG (greater greater) for append redirection, and $B (both ways) for the DOS pipe. Using the $G symbol, the following DOSKEY command correctly implements the PRINTDIR macro:

```
C:\> DOSKEY  PRINTDIR=DIR  $G  PRN <ENTER>
```

In a similar way, the macro DIRONLY uses the DOS pipe to display only the directories in the current directory:

```
C:\> DOSKEY  DIRONLY=DIR  $B  FIND  "<DIR>" <ENTER>
```

Table 27-5 summarizes DOSKEY's redirection symbols.

Symbol	Represents
$G	> DOS output redirection operator
$L	< DOS input redirection operator
GG	>> DOS append redirection operator
$B	| DOS pipe operator

Table 27-5. DOSKEY Macro I/O Redirection Symbols

Using DOSKEY's /MACROS switch, you can display the defined macros.

```
C:\> DOSKEY  /MACROS <ENTER>
SHOWEM=ECHO $1 $2 $3 $4 $5 $6 $7 $8 $9
SHOWALL=ECHO $*
TYPE=FOR %I IN ($*) DO TYPE %I
PRINTDIR=DIR $g PRN
DIRONLY=DIR $b FIND "$lDIR$g"
```

Macro Restrictions

Although DOSKEY macros are very powerful, they do have a few restrictions. First, DOSKEY stores macros in the same memory buffer it allocates for tracking your previous commands. If you create many macros, you may need to use DOSKEY's /BUFSIZE switch to increase the buffer size.

Second, one DOSKEY macro cannot invoke a second macro. For example, the following macro, NESTED, tries to invoke the macro SHOWEM:

```
C:\> DOSKEY  NESTED=SHOWEM <ENTER>
```

When you invoke NESTED, DOS will display the "Bad command" error message.

```
C:\> NESTED <ENTER>
C:\> SHOWEM
Bad command or file name
```

Third, you can only invoke DOSKEY macros from the keyboard. If you reference a macro name within a batch file, for example, DOS will not locate the command.

How DOSKEY Works

DOSKEY loads memory-resident software, which captures the DOS keyboard interrupt, letting DOSKEY examine each keystroke before the DOS command processor, COMMAND.COM, gets a chance to. In most cases, DOSKEY simply passes the keys you type along to COMMAND.COM. If you press UP ARROW or DOWN ARROW however, DOSKEY's software takes over, recalling command names on your screen. Likewise, if you press ENTER to execute a command, DOSKEY looks at the command name before anything else. If the name matches a macro, DOSKEY executes the macro's command(s). If the name does not match a macro, DOSKEY adds it to the command history and passes it along to COMMAND.COM. Because DOSKEY sees the command name before COMMAND.COM, you can redefine the internal DOS commands.

The only time DOSKEY gets to see command names is when they are typed at the keyboard. If you execute a batch file for example, DOSKEY does not get to see the command names, which explains why batch files can't successfully execute macros.

SUMMARY

DOSKEY Macros

A DOSKEY macro is similar to a single-line batch file. Like batch files, DOSKEY macros have names. You execute a DOSKEY macro by typing its name at the DOS prompt. Unlike batch files that reside on disk, DOSKEY macros reside in RAM, from which they execute very quickly. The format to create a DOSKEY macro is

```
C:\> DOSKEY Name=Command [Parameters] <ENTER>
```

Key Terms

Previous-command buffer The location in memory where the DOS 5 or later DOSKEY command stores the names of your previous commands.

Macro A sequence of characters that, when entered at the keyboard as a command, DOS replaces with one or more different commands. In DOS 5 or later, the DOSKEY command provides macro support.

Chapter 28

DOS 5 and 6 Crisis Prevention

In Chapter 22 you learned the importance of backing up the files on your hard disk to floppies. In this chapter, you will examine the DOS 5 and 6 disk "crisis-prevention" commands that let you rescue your files or disk after an inadvertent DEL or FORMAT command. Before you examine these commands keep one thing in mind: *There is no substitute for good disk backups.* Although the commands presented in this chapter may undelete files you have accidentally deleted or unformat your hard disk, the commands can't reproduce your files if your disk breaks. The DOS 5 and 6 UNDELETE and UNFORMAT commands are very powerful and very convenient. Use these two commands to complement, not replace, your disk backups.

If you are using DOS 5 or later, these crisis-prevention commands are a nice bonus. If you are not, several third-party software products exist that not only provide undelete and unformat utilities, but also other powerful utilities that will improve your system's disk and file performance.

Retrieving Deleted Files with UNDELETE

Chapter 4 introduced the DEL command. Since then you have used DEL to delete a single file or, using wildcards, a group of files. As you have learned, you must use DEL with care to prevent the accidental deletion of needed files. In Chapter 22 you learned that if you accidentally delete a file, you can restore its contents—if you have good backups. If you are using DOS 5 or later, the UNDELETE command lets you quickly undelete a recently deleted file.

To better understand how UNDELETE works, create the directory TESTDIR, and copy to the directory the DOS commands that begin with the letter D.

```
C:\> MKDIR  \TESTDIR <ENTER>
C:\> CHDIR  \TESTDIR <ENTER>
C:\TESTDIR> COPY  \DOS\D*.COM <ENTER>
```

The TESTDIR directory should contain the following:

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\TESTDIR

.            <DIR>     02-23-93  12:39a
..           <DIR>     02-23-93  12:39a
DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
        6 file(s)      33002 bytes
                    16723968 bytes free
```

Next, delete the file DOSKEY.COM.

```
C:\TESTDIR> DEL  DOSKEY.COM <ENTER>
```

A directory listing of TESTDIR shows the file has been successfully deleted.

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\TESTDIR
```

```
.            <DIR>     02-23-93  12:39a
..           <DIR>     02-23-93  12:39a
DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
       5 file(s)       27119 bytes
                    16730112 bytes free
```

To undelete the file, invoke UNDELETE, as shown here:

```
C:\TESTDIR> UNDELETE  DOSKEY.COM /DOS <ENTER>
```

The /DOS switch directs UNDELETE to only use the information that is available from DOS. Later in this chapter you will learn how to use UNDELETE's data sentry and data tracking capabilities.

To undelete a file, you must perform two steps. First, UNDELETE will display the directory entry for a file that matches the filename or wildcard specified. The filename, however, begins with a question mark, instead of the original letter, as shown here:

```
C:\TESTDIR> UNDELETE DOSKEY.COM /DOS <ENTER>

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: C:\TESTDIR
File Specifications: DOSKEY.COM
   Searching Delete Sentry control file....
    Delete Sentry control file contains    0 deleted files.

    Deletion-tracking file not found.

    MS-DOS directory contains    1 deleted files.
    Of those,   1 files may be recovered.

Using the MS-DOS directory method.

      ?OSKEY COM    5883  2-12-93   6:00a  ...A  Undelete (Y/N)?
```

If the directory entry displayed corresponds to the filename you want to undelete, type **Y**.

Next, you must tell UNDELETE the first letter of the filename you want to undelete, as shown here:

```
Please type the first character for ?OSKEY   .COM: D

File successfully undeleted.
```

UNDELETE will now undelete the file. A directory listing of TESTDIR shows DOSKEY.COM has been undeleted!

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\TESTDIR

.            <DIR>     02-23-93  12:39a
..           <DIR>     02-23-93  12:39a
DOSSHELL COM      4620 02-12-93   6:00a
DISKCOMP COM     10620 02-12-93   6:00a
DISKCOPY COM     11879 02-12-93   6:00a
DOSKEY   COM      5883 02-12-93   6:00a
        6 file(s)      33002 bytes
                    16723968 bytes free
```

Now delete all four files in the TESTDIR directory.

```
C:\TESTDIR> DEL  *.* <ENTER>
```

A directory listing of the directory reveals the directory is empty as shown here:

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\TESTDIR
```

```
.          <DIR>        02-23-93   12:39a
..         <DIR>        02-23-93   12:39a
       2 file(s)          0 bytes
                   16760832 bytes free
```

To undelete the files you have two choices. First, you can invoke UNDELETE individually for each file, or you can use a wildcard.

```
C:\> UNDELETE  *.COM /DOS  <ENTER>
```

As before, UNDELETE will display a directory entry and ask you whether or not you want to undelete the entry.

```
C:\TESTDIR> UNDELETE *.COM /DOS <ENTER>

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: C:\TESTDIR
File Specifications: *.*
    Searching Delete Sentry control file....

     Delete Sentry control file contains    0 deleted files.
     Deletion-tracking file not found.

     MS-DOS directory contains    4 deleted files.
     Of those,   4 files may be recovered.

Using the MS-DOS directory method.

        ?OSSHELL COM  4620  2-12-93  6:00a ...A  Undelete (Y/N)?
```

If you type **Y**, UNDELETE will prompt you to enter the filename's first letter.

```
Please type the first character for ?OSSHELL  .COM:
```

If you instead type N, UNDELETE will leave this entry deleted, continuing with the next file entry. To undelete all four files, respond with Y for each entry and type in the first letter of each filename.

How Can DOS Undelete a File?

When you delete a file, the file is gone. Right? Well, not exactly. When you delete a file using DEL, DOS does not actually erase the file's contents from disk right then. Instead, DOS makes the disk space that contains the file available for use by another file. When you delete one file and then create a second, DOS will very likely store the second file in one or more of the disk locations made available by the first file's deletion. When DOS stores the second file in these locations, DOS overwrites the first file's contents, just as you might record over a song on a cassette tape. Figure 28-1 illustrates this process.

If you accidentally delete a file and have not yet copied new information onto the disk, the file's original contents are still available. The UNDELETE command simply locates the file's original contents and rebuilds the file.

NOTE

If you accidentally delete a file, do not copy any information to your disk until you successfully use UNDELETE to undelete the file's contents. If you copy information to your disk, you will very likely overwrite the file's original contents, making the file impossible to undelete.

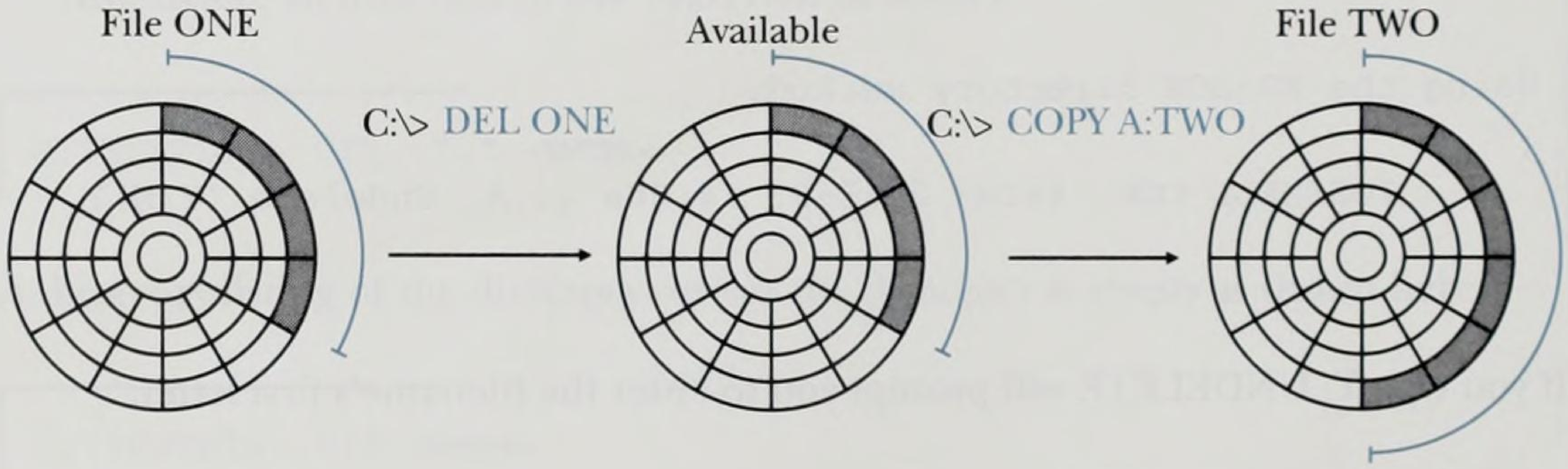


Figure 28-1. *When you delete one file and create another, DOS may overwrite the deleted file's information*

Why Doesn't DOS Know the Filename's First Letter?

A directory is a list of file entries. When you create a file, DOS adds an entry to the directory list. When you delete a file, DOS no longer displays the file's directory entry, but DOS does not actually remove the entry from the directory list. Instead, like the file's disk space, DOS marks the file's directory entry available for reuse by the next file you create. DOS makes a directory entry available for reuse by internally replacing the first letter of the filename with a special character (σ). If you delete the file DISKCOPY.COM, TESTDIR's directory entries become those shown in Figure 28-2.

When DOS searches a directory, to perform a DIR command for example, DOS ignores entries that begin with σ. UNDELETE, however, knows such entries correspond to deleted files. However, before UNDELETE can undelete the file, you need to tell UNDELETE what letter to use as the filename's first character.

Listing Files Available to Undelete

The /LIST switch directs UNDELETE to search the directory for deleted files, displaying the entries for files you can undelete. When you include /LIST, UNDELETE only lists the files. It does not undelete them. The following command directs UNDELETE to list the files available to undelete:

```
C:\TESTDIR> UNDELETE  /LIST /DOS <ENTER>
```

DOSKEY.COM	5883	03-08-91	5:05a	Attributes	Starting cluster
DOSSHELL.COM	4623	03-08-91	5:05a	Attributes	Starting cluster
DISKCOMP.COM	10652	03-08-91	5:05a	Attributes	Starting cluster
σISKCOPY.COM	11793	03-08-91	5:05a	Attributes	Starting cluster

Figure 28-2. DOS marks a directory entry as available by replacing the first character of the filename with σ.

If you have deleted all four files in the example directory, UNDELETE will display the following:

```
UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: C:\TESTDIR
File Specifications: *.*
   Searching Delete Sentry control file....
Delete Sentry control file contains     0 deleted files.

    Deletion-tracking file not found.

    MS-DOS directory contains     4 deleted files.
    Of those,    4 files may be recovered.

Using the MS-DOS directory method.

      ?OSSHELL COM      4620  2-12-93  6:00a  ...A
      ?ISKCOMP COM     10620  2-12-93  6:00a  ...A
      ?ISKCOPY COM     11879  2-12-93  6:00a  ...A
      ?OSKEY   COM      5883  2-12-93  6:00a  ...A
```

Note the two lines that describe the number of deleted files in the directory and the number that can be recovered.

```
   MS-DOS Directory contains    4 deleted files.
   Of those,  4 files may be recovered.
```

UNDELETE determines the number of deleted files in the directory by counting the number of directory entries for which the first letter of the filename is σ. Next, UNDELETE checks to make sure each file's previous contents have not been overwritten by another file. If the file's entire contents are available, UNDELETE marks the file as recoverable. In this case, UNDELETE found four files, all recoverable.

Undeleting All Available Files at One Time

As you have learned, to undelete a group of files at one time, you must individually respond with **Y** to select the file and then type in the first letter of the file's name.

UNDELETE's /ALL switch gives you a different way to undelete a group of files. When you include /ALL, UNDELETE automatically undeletes each matching file, without prompting you to confirm the file's selection or type in the file's first letter. Because UNDELETE must assign a letter to the filename's first character, UNDELETE automatically uses the # character. For example, assuming you have deleted the files in the TESTDIR directory, you can undelete all four files automatically as follows:

```
C:\TESTDIR> UNDELETE /ALL /DOS <ENTER>

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: C:\TESTDIR
File Specifications: *.*
   Searching Delete Sentry control file....
Delete Sentry control file contains    0 deleted files.

    Deletion-tracking file not found.

    MS-DOS directory contains    4 deleted files.
    Of those,    4 files may be recovered.

Using the MS-DOS directory method.

      ?OSSHELL COM     4620  2-12-93  6:00a  ...A

File successfully undeleted.

      ?ISKCOMP COM    10620  2-12-93  6:00a  ...A

File successfully undeleted.

      ?ISKCOPY COM    11879  2-12-93  6:00a  ...A

File successfully undeleted.

      ?OSKEY   COM     5883  2-12-93  6:00a  ...A

File successfully undeleted.
```

A directory listing reveals that UNDELETE has assigned the # character to the first letter of each filename.

```
C:\TESTDIR> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 1A54-45E0
 Directory of C:\TESTDIR

.            <DIR>     02-23-93  12:39a
..           <DIR>     02-23-93  12:39a
#OSSHELL COM      4620 02-12-93   6:00a
#ISKCOMP COM     10620 02-12-93   6:00a
#ISKCOPY COM     11879 02-12-93   6:00a
#OSKEY   COM      5883 02-12-93   6:00a
        6 file(s)      33002 bytes
                    16723968 bytes free
```

After the files are undeleted, you can use the DOS RENAME command to assign the original filenames.

Because each filename in a directory must be unique, UNDELETE will use one of the following characters for the filename's first character until the filename becomes unique:

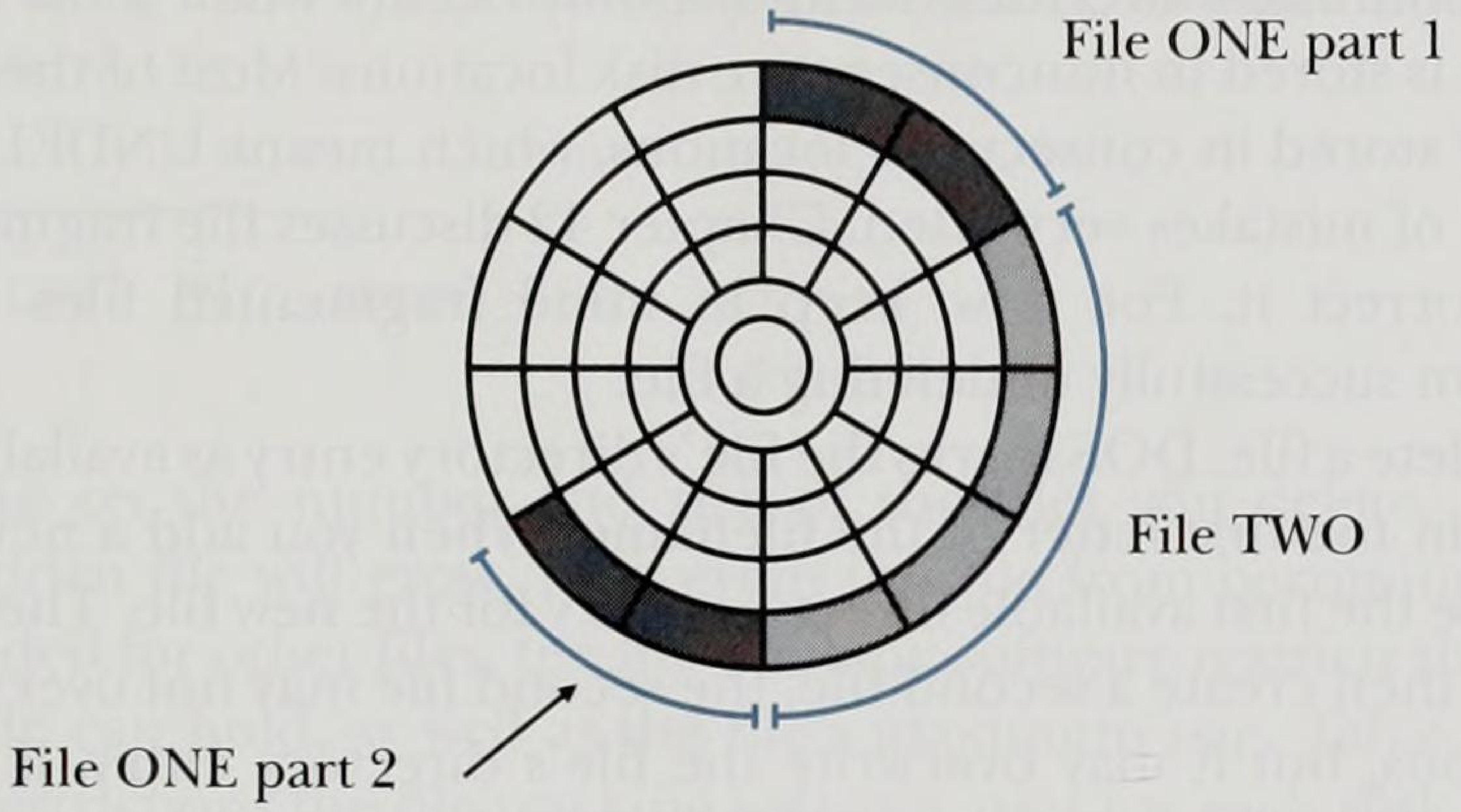
#%$-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ.

The UNDELETE command supports additional command-line switches you will examine later in this chapter, following the discussion of the MIRROR command.

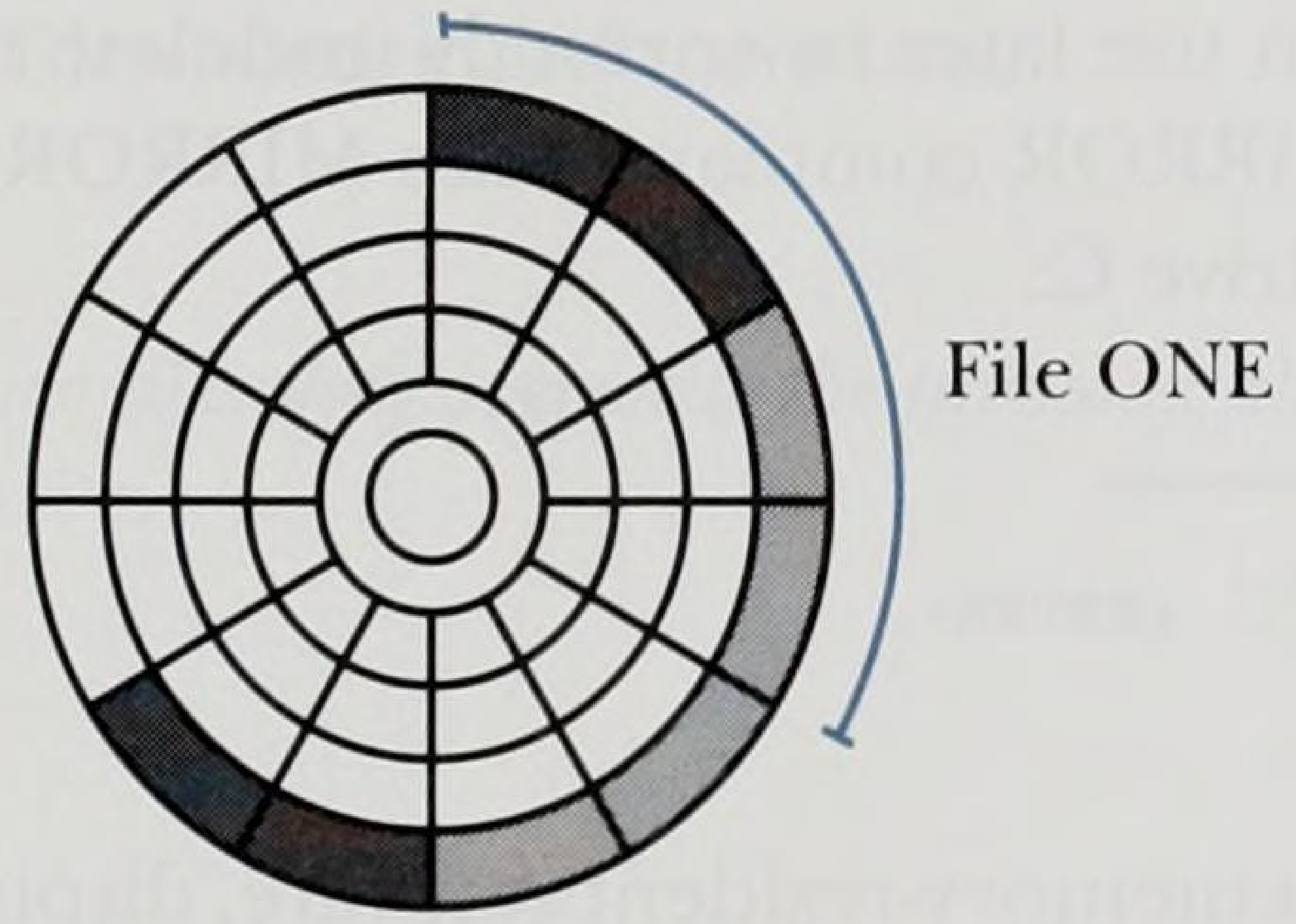
Protecting Your Disk and Files with File Tracking

If you are using DOS 5, the MIRROR command is a memory-resident program that stores information UNDELETE can use to correctly undelete files. In addition, MIRROR stores information that assists UNFORMAT in unformatting your hard disk following an accidental FORMAT command. If you are using DOS 6, DOS has integrated MIRROR's file-tracking capabilities into the UNDELETE command. To understand how file tracking assists UNDELETE, you must understand how UNDELETE is sometimes prevented from undeleting a file.

The following illustration shows two files named ONE and TWO. As you can see, file ONE is stored on disk in nonconsecutive storage locations.



Assume you accidentally delete both files. When you invoke UNDELETE to undelete file ONE, UNDELETE can determine from the file's size that file ONE requires two storage locations. UNDELETE can also determine, from the file's directory entry, where the file's first location resides on disk. Unfortunately, UNDELETE doesn't know where the file's second location is. UNDELETE assumes the second location immediately follows the first, resulting in the file containing the disk locations shown here:



Because UNDELETE has assumed file ONE's storage locations were consecutive, it has incorrectly merged file ONE's contents with what used to be those of file TWO.

If you examine file ONE's contents using TYPE or your word processor, you will very likely see the error. Unfortunately, because UNDELETE has assigned file TWO's previous contents to file ONE, UNDELETE now thinks file TWO's contents have been overwritten. If you try to undelete file TWO, UNDELETE will display a message telling you it can't undelete the file.

```
Starting cluster is unavailable. This file cannot be recovered
with the UNDELETE command. Press any key to continue.
```

Further, because UNDELETE undeleted enough disk locations to match file ONE's size, UNDELETE won't let you recover the other disk location that actually belongs to file ONE.

Fortunately, confusion over disk locations only occurs when a file is *fragmented*, meaning the file is stored in nonconsecutive disk locations. Most of the files on your disk are typically stored in consecutive locations, which means UNDELETE doesn't make these kind of mistakes very often. Chapter 42 discusses file fragmentation and ways you can correct it. For now, keep in mind fragmented files may prevent UNDELETE from successfully undeleting a file.

When you delete a file, DOS marks the file's directory entry as available by placing the character σ in the first letter of the filename. When you add a new file to your disk, DOS will use the first available directory entry for the new file. Therefore, if you delete a file and then create a second file, the second file may not overwrite the first file's disk locations, but it may overwrite the file's directory entry. If UNDELETE cannot locate a file's directory entry, UNDELETE cannot undelete the file, even though all of the contents may still be on disk. As you will learn in this section, file tracking directs DOS to save the directory entry and disk locations of files that you delete from your disk.

File tracking assists UNDELETE by tracking the actual disk locations used by each file you delete from your disk. When you invoke MIRROR or the DOS 6 UNDELETE command using the /T switch, they create a hidden file in the root directory. When you delete a file from the disk, the file-tracking software stores specifics about the file that UNDELETE can use later to correctly undelete the file.

The following MIRROR command uses MIRROR's /T switch to install file-deletion tracking from drive C:

```
C:\> MIRROR  /TC <ENTER>
```

MIRROR will load its memory-resident software, displaying the following message on your screen:

```
Creates an image of the system area.

Drive C being processed.

The MIRROR process was successful.
Deletion-tracking software being installed.
The following drives are supported:
Drive C - Default files saved.

Installation complete.
```

If you are using DOS 6, the following UNDELETE command loads file-tracking software for drive C:

```
C:\> UNDELETE /TC <ENTER>
```

Depending on the number and size of the files you delete, the file-tracking software's hidden file will grow. To prevent this file from becoming so large it uses up space needed for other files, the file-tracking software restricts the number of file entries the file can hold, as well as the file's maximum size. Table 28-1 defines the default size restrictions the file-tracking software uses for each disk type.

The file-tracking software does not store the file's original information, just the disk location of the information. Using this information, UNDELETE can successfully undelete fragmented files.

NOTE

If you delete enough files to reach either the maximum number of entries or the maximum file size, the file-tracking software will discard the oldest information in the file to make room for new file-deletion information.

If you copy files to your disk, the new files may overwrite the previous files' disk locations even though file tracking is active.

NOTE

Disk Type	Maximum Entries	Maximum Size
360Kb	25	5Kb
720Kb	50	9Kb
1.2Mb	75	14Kb
1.44Mb	75	14Kb
20Mb	101	18Kb
32Mb	202	36Kb
Over 32Mb	303	55Kb

Table 28-1. *Maximum Number and File Restrictions for the Deletion-Tracking File*

If you find the number of entries MIRROR tracks in the delete file to be insufficient, you can specify a value up to 999 when you first invoke MIRROR. For example, the following command loads file-deletion tracking for drive C to support 500 entries:

```
C:\> MIRROR /TC-500 <ENTER>
```

Because MIRROR loads memory-resident software, you must either restart DOS or unload MIRROR from memory with the /U switch if you want to change one of MIRROR'S settings.

If you are using DOS 6, the following UNDELETE command installs file-tracking support for 500 entries:

```
C:\> UNDELETE /TC-500 <ENTER>
```

Revisiting UNDELETE

When you use file tracking, UNDELETE has two ways to locate files. First, UNDELETE can search the directory for a matching entry and rebuild the file, assuming the file's contents are in consecutive locations. Second, and ideally, UNDELETE will use the information in the file-tracking file. By default, UNDELETE uses the deletion-tracking information, if it is available, as well as the information UNDELETE can determine by reading the directory. If you use the /LIST switch, UNDELETE lists the entries it can undelete without using the file-tracking file, as well as those entries for which it has deletion-tracking information.

```
C:\TESTDIR> UNDELETE /LIST <ENTER>

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: C:\TESTDIR
File Specifications: *.*
   Searching Delete Sentry control file....
    Delete Sentry control file contains    0 deleted files.
   Searching Delete Tracker file....
Deletion-tracking file contains    4 deleted files.
```

```
    Of those,   4 files have all clusters available,
                0 files have some clusters available,
                0 files have no clusters available.

    MS-DOS directory contains    4 deleted files.
    Of those,   4 files may be recovered.

Using the Delete Sentry method.

   Searching Delete Sentry control file....
No entries found.
```

Depending on the files listed, there may be times you want UNDELETE only to undelete those files it knows about using DOS. To do so, invoke UNDELETE using the /DOS switch.

```
C:\> UNDELETE  *.*  /DOS <ENTER>
```

Likewise, there may be times you want to only undelete those files for which file tracking is active. To do so, invoke UNDELETE with the /DT switch, and UNDELETE will undelete only those files listed in the file-tracking file.

```
C:\> UNDELETE  *.*  /DT <ENTER>
```

DOS 6 Data Sentry Protection

In DOS 6, UNDELETE provides a third way to recover deleted files called the data sentry. When you use UNDELETE's data sentry, UNDELETE copies each file that you delete to a directory called SENTRY. By making a second copy of the file, another file cannot later overwrite the deleted file on disk, making the file's undeletion impossible. To install data sentry protection for drive C, invoke UNDELETE using the /S switch, as shown here:

```
C:\> UNDELETE /S:C <ENTER>
```

Because data sentry protection makes a copy of every file delete, it can consume considerable disk space. If you are sure that you don't need to undelete any files, you can get rid of the sentry's file copies using the /PURGE switch, shown next.

```
C:\> UNDELETE /PURGE <ENTER>
```

Saving Disk Information for UNFORMAT Using the DOS 5 MIRROR Command

As briefly discussed, the DOS 5 MIRROR command supports both UNDELETE and UNFORMAT. Each time you invoke MIRROR, it records information about your disk's root directory, boot record, and file allocation tables in the root directory file MIRROR.FIL. Using this information, the UNFORMAT command can recover all or a portion of your disk following an accidental FORMAT command.

NOTE *If you create or change files after running the MIRROR command, UNFORMAT may not fully recover your disk in the event of an accidental disk format. The more often you invoke MIRROR, the better your chances of UNFORMAT being able to fully recover your disk. At a minimum, you should invoke MIRROR each time your system starts by placing the MIRROR command in your AUTOEXEC.BAT file. If you want MIRROR to track file deletions (and most users should) invoke MIRROR from within AUTOEXEC.BAT, and be sure to include the /T switch.*

Recovering a Disk with UNFORMAT

In Chapter 7 you learned that formatting a disk destroys the disk's current contents. You may, however, be able to recover all or part of the disk's contents following an accidental format by using the UNFORMAT command.

NOTE *UNFORMAT is a recovery command you should invoke only after an accidental disk format. Invoking UNFORMAT on a disk that does not need recovery may result in the loss of information. Do not execute the UNFORMAT commands presented in this section. Instead, examine the commands and note their functionality. You can refer to UNFORMAT any time in the future should an accident occur. For now, just execute the commands presented in the "Hands On" section.*

As was the case with UNDELETE, UNFORMAT can recover your disk using information it gets strictly from DOS or information contained in the DOS 5 MIRROR file MIRROR.FIL. If you are using DOS 6, DOS does not provide the MIRROR command. As a result, UNFORMAT must rebuild the disk without help from the MIRROR file MIRROR.FIL. One of the first problems users encounter when they accidentally format their hard disk is trying to find a floppy disk containing the

UNFORMAT command. Before you continue reading, take out an unused floppy disk and format the floppy as a bootable floppy disk.

```
C:\> FORMAT  A:  /S <ENTER>
```

When FORMAT completes, copy the file UNFORMAT.COM from the DOS directory to the floppy disk.

```
C:\> COPY  \DOS\UNFORMAT.COM  A: <ENTER>
```

Label the disk as UNFORMAT and place the floppy disk in a safe location. Should you ever need to recover your disk, you can use the floppy disk to do so.

Unformatting a Disk Using the DOS 5 MIRROR Command

If you use the MIRROR file MIRROR.FIL, UNFORMAT can only recover your disk to the last saved state. UNFORMAT's /J switch directs UNFORMAT to examine your disk and verify that the information on your disk matches the information in the file MIRROR.FIL. The /J switch does not UNFORMAT your disk; it simply compares the disk to MIRROR's file.

```
C:\> UNFORMAT  C:  /J <ENTER>
```

UNFORMAT will display a screenful of warning messages followed by the line "Just checking this time." to let you know an unformat operation will not take place.

```
Restores the system area of your disk by using the image file
created by the MIRROR command.

   WARNING !!          WARNING !!
This command should be used only to recover from the inadvertent
use of the FORMAT command or the RECOVER command. Any other use
of the UNFORMAT command may cause you to lose data! Files
modified since the MIRROR image file was created may be lost.

Searching disk for MIRROR image.

Just checking this time. No changes written to disk.
```

The MIRROR command keeps two copies of your disk's system information, the latest copy (MIRROR.FIL) and the previous copy (MIRROR.BAK). Each time you invoke the MIRROR command, MIRROR renames the existing file MIRROR.FIL to MIRROR.BAK and copies the current disk information to the file MIRROR.FIL. When you invoke the UNFORMAT command, UNFORMAT locates these two files, displaying each file's creation date on your screen. UNFORMAT then asks you which file you want to use.

```
The last time the MIRROR or FORMAT command was used
was at 19:32 on 04-10-91.
The prior time the MIRROR or FORMAT command was used
was at 11:59 on 04-10-91.

If you wish to use the last file as indicated
above, press L. If you wish to use the prior
file as indicated above, press P. Press ESC
to cancel UNFORMAT.
```

If you want to use the latest copy, type **L**. To use the previous copy, type **P**. To exit UNFORMAT, press ESC.

If you are using UNFORMAT's /J switch to compare the disk's contents to the MIRROR file, and both match, UNFORMAT will display the following message:

```
The system area of drive C has been verified
to agree with the MIRROR image file.
```

If the disk's current contents don't agree with the MIRROR file, UNFORMAT will display the following message:

```
The system area does not agree with the
MIRROR image file.
```

If this message appears, invoke MIRROR to bring MIRROR's image file up to date.

If you need to UNFORMAT your hard disk and have used MIRROR on a regular basis, place the floppy disk you previously created that contains UNFORMAT into drive A and issue the following command:

```
A:\> UNFORMAT  C: <ENTER>
```

As before, UNFORMAT will display its warning messages and prompt you to select the latest or previous version of the MIRROR file, or to press ESC to end the UNFORMAT operation.

```
Restores the system area of your disk by using the image file cre-
ated by the MIRROR command.

  WARNING !!          WARNING !!

This command should be used only to recover from the inadvertent
use of the FORMAT command or the RECOVER command. Any other use
of the UNFORMAT command may cause you to lose data!  Files
modified since the MIRROR image file was created may be lost.

Searching disk for MIRROR image.

The last time the MIRROR or FORMAT command was used
was at 19:32 on 04-10-91.
The prior time the MIRROR or FORMAT command was used
was at 11:59 on 04-10-91.

If you wish to use the last file as indicated
above, press L. If you wish to use the prior
file as indicated above, press P. Press ESC
to cancel UNFORMAT.
```

If you type either **L** or **P**, UNFORMAT will validate the MIRROR file, making sure the file is complete (although not necessarily current). UNFORMAT then asks you if you really want to update your disk's system area.

```
The MIRROR image file has been validated.

Are you sure you want to update the system area of
  your drive C (Y/N)?
```

To perform the UNFORMAT operation type **Y**. To end the operation type **N**. If you continue the operation, UNFORMAT will rebuild your disk displaying the messages shown in the following example.

```
The system area of drive C has been rebuilt.

You may need to restart the system.
```

Perform a directory operation to examine your disk's root directory. Next, invoke CHKDSK to examine your disk's current structure. If CHKSDK reports several errors, invoke CHKDSK again, using the output redirection operator to write CHKDSK's output to your printer. The files CHKDSK lists may be damaged. You will need to restore the files from your backup disks. Invoke CHKSDK using the /F switch to correct the errors.

Unformatting a Disk Without MIRROR's Support

If you have never used MIRROR, or have not used it in a long time, you may be better off letting UNFORMAT rebuild your disk using the information it can get from DOS and ignoring the MIRROR file. Likewise, if you are using DOS 6, which does not provide the MIRROR command, you can only use the information known to DOS. UNFORMAT's /TEST switch lets you display the directories UNFORMAT will rebuild if UNFORMAT only uses information available from DOS. Like the /J switch, the /TEST switch only displays information to your screen about the unformat process, it does not actually unformat your disk. The following command uses UNFORMAT's /TEST switch to examine the directories UNFORMAT can rebuild:

```
A:\> UNFORMAT  C:  /TEST <ENTER>
```

In this case, UNFORMAT will display a warning message telling you that unformatting without the MIRROR file may not completely recover your disk.

```
 CAUTION !!

This attempts to recover all the files lost after a
format, assuming you've not been using the MIRROR command. This
method cannot guarantee complete recovery of your files.
```

Remember, UNFORMAT is a *crisis* recovery program. If you accidentally format your disk, be grateful for any files UNFORMAT can recover—partial disk recovery is better than none.

Next, UNFORMAT will display additional messages describing the operation and a prompt asking you if you want to continue:

```
The search-phase is safe: nothing is altered on the disk.
You will be prompted again before changes are written to the
disk.


Using drive C:

Are you sure you want to do this?
If so, press Y; anything else cancels.
```

Type **Y** to continue the text. UNFORMAT will display the message "Simulation only." letting you know it will not unformat the disk.

```
Simulation only.
```

Next, UNFORMAT will display a count of the number of files and directories it has located in the root.

```
Files found in the root: nn
Subdirectories found in the root: nn
```

UNFORMAT now begins searching your disk for directories, displaying the percentage of your disk it has examined as well as the number of directories it has found.

```
Searching disk...
 nn% searched, nn subdirectories found.
```

When UNFORMAT completes its search, it will display the directory paths of each directory it encountered, followed by a count of the number of files found.

```
Walking the directory tree to locate all files...
Path=C:\
Path=C:\DOS\
Path=C:\
Path=C:\BATCH\
  :     :
```

```
Path=C:\WP\LETTERS\

Files found: nnnn
```

UNFORMAT will now display a prompt asking you if you want to continue.

```
Simulation only.

Are you sure you want to do this?
If so, press Y; anything else cancels.
```

Type **Y** to continue, and UNFORMAT will list the fragmented files it encountered, telling you it can only recover a portion of the file. UNFORMAT will then prompt you to specify whether you want to delete the file or truncate (reduce) the file's size to only the recoverable information. To delete the file type **D**, and to truncate the file type **T**. Because the /TEST switch results in a *simulation* of the disk-rebuild process, you can type **D** or **T** for each filename displayed without actually affecting the file.

```
FILENAME.EXT nnn 12-25-90 8:30pm ONLY nnn bytes are recoverable
Truncate or Delete this file?
```

After you respond to the delete or truncate prompt for each file, UNFORMAT will display the number of files it can recover, followed by a message indicating that the operation is complete.

```
nnnn files recovered.

Operation completed.
```

If you want a hardcopy listing of the information UNFORMAT displays to your screen, invoke UNFORMAT with the /P switch.

```
A:\> UNFORMAT  C:  /TEST  /P <ENTER>
```

If you have never invoked the MIRROR command, UNFORMAT will automatically rebuild your disk using only the information it can obtain from DOS. If your disk contains an old MIRROR file, you need to tell UNFORMAT not to use the file by invoking UNFORMAT with the /U switch.

```
A:\> UNFORMAT  C:  /U <ENTER>
```

The /U switch directs UNFORMAT to perform a "DOS only" unformat operation that follows the exact same steps as the unformat operation shown using /TEST. This time however, UNFORMAT will update the disk.

Safe Versus Unsafe Format Operations

In most versions of DOS, the FORMAT command, when formatting a hard disk, does not actually overwrite the contents of the disk it is formatting. Instead, FORMAT simply initializes the disk's root directory and file allocation tables such that DOS considers the disk empty. Actually, the disk's previous information is still on disk, which explains how UNFORMAT can recover much of your disk following an accidental format operation. Because such format operations don't actually scrub the data off your disk, the operations are called *safe formats.*

In DOS 5, FORMAT lets you perform safe formats on your hard disk and floppy disks. By default, each time you format a disk containing data, FORMAT attempts to record, on the disk, information similar to that recorded by MIRROR that UNFORMAT can use to recover the disk. Note the message "Saving UNFORMAT information" in the output of the FORMAT command shown here:

```
C:\> FORMAT  A: <ENTER>
Insert new diskette for drive A:
and press ENTER when ready...

Checking existing disk format.
Saving UNFORMAT information.
Verifying 360K
 nn percent of disk formatted
```

In most cases, you will want to perform safe formats. Not only do safe formats give you a chance to recover your information, they also execute faster than unsafe format operations because FORMAT only examines a small percentage of the disk.

In some cases you may have information on the disk you are formatting that you don't want another user to unformat and view. Using the FORMAT command's /U switch (DOS 5), you can direct FORMAT to completely overwrite a disk's existing data, making an unformat operation impossible. For more information on safe and unsafe format operations, see FORMAT in the Command Reference of this book.

SUMMARY

UNDELETE

The UNDELETE command lets you undelete one or more accidentally deleted files. The format of the UNDELETE command is as follows:

```
C:\> UNDELETE FILENAME.EXT <ENTER>
```

NOTE

If you accidentally delete one or more files, do not copy any files to or store any information on your disk until you use UNDELETE to undelete the file. If you overwrite the file's contents, UNDELETE cannot recover the file.

File-Deletion Tracking

The DOS 5 MIRROR command installs memory-resident software that stores, in a root directory file, the directory entry and the disk locations of each file you delete from disk. If you are using DOS 6, file tracking is built into UNDELETE and the MIRROR command is not provided. Using this information, UNDELETE has a better chance of correctly undeleting a file.

NOTE

To provide deletion tracking, memory-resident software must perform additional processing each time you delete a file. Depending on the size and number of files you are deleting from your disk, you may notice the overhead when DEL takes longer to complete.

Using MIRROR to Assist UNFORMAT

The MIRROR command stores information in the root directory file MIRROR.FIL that UNFORMAT can use to recover your disk following an accidental disk format. Specifically, MIRROR stores the current state of the disk's root directory and file allocation table. To improve the chances of UNFORMAT fully recovering your disk, you need to invoke MIRROR on a regular basis.

UNFORMAT

The UNFORMAT command rebuilds a disk following an accidental disk format. If the file is present, UNFORMAT will use MIRROR.FIL (created by MIRROR) to rebuild your disk. If the file is not current, UNFORMAT can only

SUMMARY

recover a portion of your files. If the file does not exist, UNFORMAT will rebuild your disk using only information it can get from DOS. Using only DOS information to rebuild a disk is the least reliable method.

NOTE

The best defense against an accidental disk format is regular disk backup (see Chapter 22).

Hands On

If you have not already done so, edit your AUTOEXEC.BAT file and include either the command MIRROR to save your disk's systems information or MIRROR /TC to also enable file-deletion tracking on drive C. Next, invoke MIRROR from the keyboard.

```
C:\> MIRROR /TC <ENTER>
```

Using MKDIR, create the directory TEST. Use CHDIR to select TEST.

```
C:\> MD \TEST <ENTER>
C:\> CD \TEST <ENTER>
```

Next, copy the files with the SYS extension from your DOS directory to TEST.

```
C:\TEST> COPY \DOS\*.SYS \TEST <ENTER>
```

Using DIR, display the directory's contents.

```
C:\TEST> DIR <ENTER>
```

Delete the files beginning with the letter A.

```
C:\TEST> DEL  A*.* <ENTER>
```

Invoke UNDELETE with /LIST and /DOS to display the recoverable files.

```
C:\TEST> UNDELETE  /LIST /DOS <ENTER>
```

Use the following command to undelete the files:

```
C:\TEST> UNDELETE  *.* /DOS <ENTER>
```

Next, delete all the files in the directory.

```
C:\TEST> DEL  *.* <ENTER>
```

Use UNDELETE's /All switch to undelete all the files.

```
C:\TEST> UNDELETE  /ALL /DOS <ENTER>
```

If you have MIRROR's file-deletion tracking enabled, UNDELETE will automatically undelete the files using each file's correct name. If not, UNDELETE will undelete the files using the # character for the first letter of the filename.

Use DEL to delete the TEST directory's files and RMDIR to remove the directory. Then use the FORMAT command to format an unused floppy disk and create the following directories on the floppy with MKDIR.

```
C:\> MD  A:ONE <ENTER>
C:\> MD  A:TWO <ENTER>
C:\> MD  A:THREE <ENTER>
```

Use DIR to display the floppy disk's contents.

```
C:\> DIR  A:
```

Next, format the floppy disk, overwriting the disk's current contents.

```
C:\> FORMAT  A: <ENTER>
```

When FORMAT completes, use DIR to verify that the disk is empty.

```
C:\> DIR  A: <ENTER>
```

Next, invoke UNFORMAT to rebuild the floppy disk.

```
C:\> UNFORMAT  A: <ENTER>
```

A directory listing of the disk will reveal the disk's original directories.

```
C:\> DIR  A: <ENTER>
```

Review

1. What two steps does DEL perform when you delete a file?
2. What is file-deletion tracking?
3. How does MIRROR assist UNFORMAT?
4. When would you unformat a disk using only DOS information, as opposed to information in the file MIRROR.FIL?
5. What is stored in the file MIRROR.BAK?

6. What is the best defense against accidental file deletions or disk formats?
7. How do you load file-deletion tracking under DOS 6?
8. What is DOS 6 data sentry protection?

Answers

1. When you delete a file, DEL marks the disk locations that contained the file's contents as available for use by another file. Next, DEL marks the file's directory entry as available for reuse by assigning the character σ to the first letter in the filename. DEL does not actually erase the file's contents from your disk.
2. The MIRROR command provides file-deletion tracking, which records information about each file you delete, such as the file's directory entry and disk locations, in a hidden file in your disk's root directory. Using this information, UNDELETE is more likely to correctly undelete a file.
3. The MIRROR command records key information about your disk (the boot record, root directory, and file allocation tables) that UNFORMAT can use to rebuild your disk after an accidental FORMAT command. MIRROR places this information in the root directory file MIRROR.FIL.
4. If you have not invoked the MIRROR command in a long time, the information in the file MIRROR.FIL will be very obsolete. Therefore, you are probably better off letting UNFORMAT attempt to recover the disk using DOS data.
5. Each time you invoke MIRROR, it stores the disk's current boot record, root directory, and file allocation tables in the file MIRROR.FIL. In addition, MIRROR renames the previous contents of MIRROR.FIL as MIRROR.BAK (BAK stands for backup), giving you a previous and current copy of the information UNFORMAT needs to recover your disk.
6. The best defense against accidental file deletions and disk formats is current backup disks.

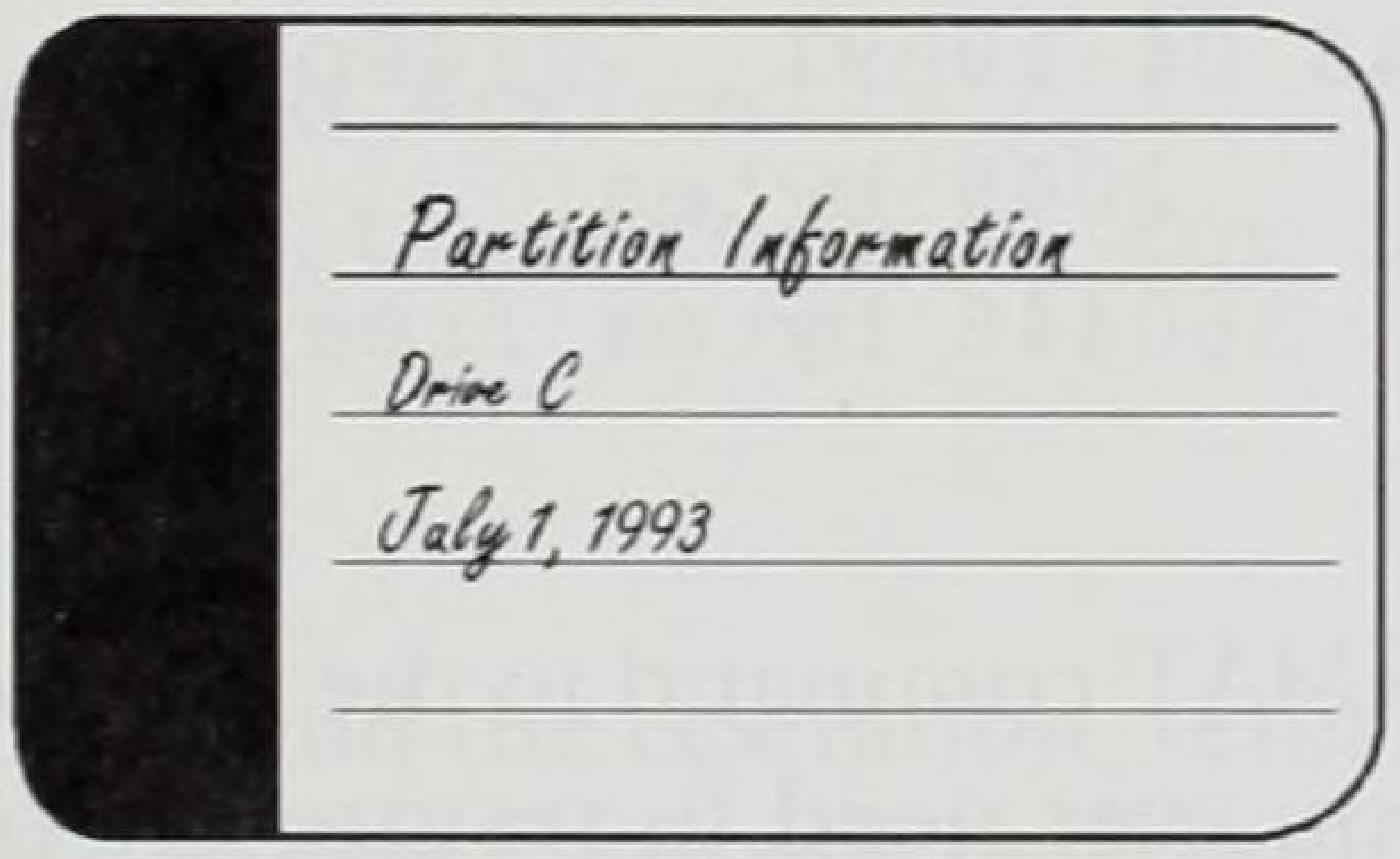
7. DOS 6 does not provide a MIRROR command. To install file-tracking under DOS 6, invoke UNDELETE using the /T switch.
8. DOS 6 data sentry protection directs DOS to copy every file that you delete to a directory named SENTRY. If you later need to undelete a file, UNDELETE can use the copy.

Advanced Concepts

FDISK records your disk's partition information, beginning at the first sector of your hard disk. This sector is called the master boot record. Each time you start your computer, the computer reads the master boot record to determine from which partition it should boot the operating system. Most DOS commands or programs cannot access the master boot record. However, if your computer encounters a virus or you accidentally change the master boot record using FDISK, you can lose all the information the disk contains because the information describing the partition boundaries has been lost.

Using the DOS 5 MIRROR command, however, you can save the master boot record to a floppy disk. Should the partition table become damaged or corrupted, you can use UNFORMAT to restore the partition table information from floppy.

To begin, format an unused floppy disk and label the disk "Partition Information" (be sure to include the drive letter and date) as shown here:



Next, invoke the MIRROR command using the /PARTN switch.

```
C:\> MIRROR  /PARTN <ENTER>
```

MIRROR will display a message explaining that it will copy the master boot record to the file PARTNSAV.FIL on the floppy disk and then prompt you for the drive letter of the target disk, displaying drive A as the default.

```
Disk Partition Table saver.

The partition information from your hard drive(s) has been read.

Next, the file PARTNSAV.FIL will be written to a floppy disk.
Please insert a formatted diskette and type the name of the
    diskette drive.
What drive? A
```

Press ENTER to copy the partition table information to the floppy disk. When MIRROR completes, a directory listing of drive A reveals the file PARTNSAV.FIL.

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 313C-1AF3
 Directory of A:\

PARTNSAV FIL       2048 04-10-91    7:48p
       1 file(s)        2048 bytes
                      360448 bytes free
```

As before, copy the UNFORMAT command to the floppy disk. Place the floppy disk in a safe location. Should you ever need to restore the partition table information, invoke UNFORMAT with the /PARTN switch.

```
A>  UNFORMAT  /PARTN <ENTER>
```

UNFORMAT will display a prompt asking you to type the letter of the floppy disk containing the file PARTNSAV.FIL. UNFORMAT displays the drive letter A as the default.

```
A> UNFORMAT  /PARTN <ENTER>
Hard Disk Partition Table restoration.

Insert the diskette containing the file PARTNSAV.FIL
and type the letter of that disk drive.
What drive? A
```

After it reads PARTNSAV.FIL, UNFORMAT will display the date and time the partition table information was created, as well as information about your hard disks.

```
Partition information was saved by Mirror, 4-10-91  7:48pm

Old partition info for fixed disk #1 (DL=80h):

             Total_size    Start_partition   End_partition
  Type     Bytes Sectors Cyl Head Sector  Cyl Head Sector   Rel#
--------- ----------- ------------- ------------- ------
HUGE Boot  191M 391168  0  1   1   814    14   32    32
```

Next, UNFORMAT will display your two options.

```
Options: Q - quit, take no action.
         1- restore partition records for fixed disk #1.
Which option?
```

If you type Q, UNFORMAT will end the operation, returning to the DOS prompt. If you type 1, UNFORMAT will display a second prompt to verify that you really want to update the partition table.

```
Are you sure you want to do this?
If so, press Y; anything else cancels.
```

If you type Y, UNFORMAT will update the partition table and ask you to insert a bootable floppy disk in drive A.

```
Operation completed.

Insert a DOS boot disk in drive A and press ENTER to reboot....
```

Depending on the severity of the disk error that damaged your disk partition, you may not be able to boot from your hard disk. By booting DOS from floppy, you can use DIR and CHKDSK to examine your hard disk once your system successfully starts.

NOTE

If the partition table information stored on the file PARTNSAV.FIL does not correspond to your hard disk size, UNFORMAT will not use the file.

Key Terms

Deletion tracking Information tracked by the DOS 5 MIRROR command about each file you delete from your disk.

Undelete The process of rebuilding a file on disk previously deleted by DEL.

Unformat The process of rebuilding a disk that has been accidentally formatted.

Chapter 29

Using DEBUG

A program is a list of instructions that tells your computer what to do. When you execute a program, your computer performs each instruction, beginning with the first and ending with the last. Because programs are written by humans, who periodically make mistakes, the list of computer instructions isn't always correct. When this occurs, the errors may cause the computer to behave incorrectly. These errors are not the computer's fault. The computer is simply performing the instructions given. Computer programmers refer to program errors as *bugs.* The DEBUG command is a tool to help programmers remove bugs from their programs. The process of locating and removing program errors is called *debugging.*

Although the DEBUG command began as a programmer's tool, it is most used today by end users to build simple commands. Programmers now use more powerful debuggers to assist them in locating program errors.

As you know, files with the EXE or COM extensions contain commands. These commands were created by programmers using programming languages such as C or Pascal. Using DEBUG you can create your own simple commands, even if you know nothing about programming. In this chapter, you will create several commands using DEBUG.

Creating Simple COM Commands

DEBUG lets you create COM files containing simple commands. To create a command, you need to provide DEBUG with the command's instructions. This chapter provides the instructions you'll need for each command. If you pick up almost any computer magazine, you'll find the instructions for other commands you can use DEBUG to create.

DEBUG is an external command in the DOS directory. The following example uses DEBUG to create a command named HELLO.COM that displays the message "Hello, world!" on your screen. To begin, invoke DEBUG with the filename HELLO.COM, as shown here:

```
C:\> DEBUG  HELLO.COM  <ENTER>
File not found
```

Because the file HELLO.COM does not yet exist, DEBUG displays the "File not found" message. DEBUG uses the hyphen as its command prompt. DEBUG commands are each one letter, possibly followed by additional parameters, such as a filename or memory location. To tell DEBUG you want to enter the command's instructions, type the command **A 100** as shown below. Note that a hyphen remains at the beginning of the line with the command.

```
C:\> DEBUG  HELLO.COM  <ENTER>
File not found

-A 100
16C9:0100
```

DEBUG responds to your command by displaying a set of eight hexadecimal (base 16) numbers, separated by a colon. These numbers represent memory locations. You can ignore the first four numbers, which, depending on your DOS version, CONFIG.SYS entries, and memory-resident programs, will probably differ from those shown here. Pay attention to the four numbers following the colon. These four numbers should match the numbers shown in the examples. Because DEBUG was originally designed as a tool for programmers, DEBUG displays the memory locations as hexadecimal values.

DEBUG is now ready for you to type in the program's instructions. You will enter the instructions using symbols called *assembly language.* Because this chapter provides the program instructions, you don't need to understand assembly language itself. You can leave that chore to experienced programmers.

To begin, type the assembly language instruction **MOV AH, 9** and press ENTER.

```
16C9:0100 MOV AH, 9  <ENTER>
16C9:0102
```

DEBUG will display a second memory address. Type in the instruction **MOV DX, 10B** as shown here:

```
16C9:0100   MOV  AH, 9      <ENTER>
16C9:0102   MOV  DX, 10B   <ENTER>
```

Repeat this process until you have typed in the following instructions.

```
16C9:0100   MOV  AH, 9      <ENTER>
16C9:0102   MOV  DX, 10B    <ENTER>
16C9:0105   INT  21         <ENTER>
16C9:0107   MOV  AH, 4C     <ENTER>
16C9:0109   INT  21         <ENTER>
16C9:010B   DB  'Hello, world!$' <ENTER>
16C9:0119
```

You have now finished entering the instructions. Press ENTER, and DEBUG will redisplay its hyphen prompt.

```
16C9:0100   MOV  AH, 9      <ENTER>
16C9:0102   MOV  DX, 10B    <ENTER>
16C9:0105   INT  21         <ENTER>
16C9:0107   MOV  AH, 4C     <ENTER>
16C9:0109   INT  21         <ENTER>
16C9:010B   DB  'Hello, world!$' <ENTER>
16C9:0119   <ENTER>
-
```

You must now tell DEBUG the number of bytes of memory the assembly language instructions consume. To do so, type the command **R CX**. Remember, a hyphen stays at the beginning of the command line.

```
-R CX
```

DEBUG stores the command's length in a storage location called CX. The R command directs DEBUG to display CX's current value, and then to prompt you using a colon, to type in the new value. To determine the command's size, subtract the command's starting address from its ending address. In this case, the ending address is 0119 and the starting address is 0100. The command's size becomes 19 (0119-0100). Type **19** and press ENTER.

```
CX 0000
:19  <ENTER>
```

Next, use DEBUG's W command to write your command out to disk, storing it in the file HELLO.COM.

```
-W  <ENTER>
Writing 00019 bytes
```

Use the Q command to quit DEBUG, returning to the DOS prompt.

```
-Q  <ENTER>

C:\>
```

You can now execute your newly created command.

```
C:\> HELLO  <ENTER>
Hello, world!
```

If your command did not execute successfully, you may need to restart your computer using the CTRL-ALT-DEL key combination. Next, delete the file HELLO.COM and invoke DEBUG as before, typing in the DEBUG commands and assembly language instructions very carefully.

When you create a command using DEBUG, follow these steps:

1. Invoke DEBUG with the command name (C:> DEBUG *FILENAME*.COM).
2. Enter the instructions using the DEBUG command A 100.
3. Type in the assembly language instructions, pressing ENTER to return to DEBUG's hyphen prompt following the last instruction.
4. Use DEBUG's R CX command to tell DEBUG the command's size.
5. Use DEBUG's W command to write the command to disk.
6. Use DEBUG's Q command to quit DEBUG and return to the DOS prompt.

Using these six steps, the following example creates a command named BEEP.COM that sounds your computer's built-in speaker one time.

```
C:\> DEBUG BEEP.COM  <ENTER>
File not found

-A 100 <ENTER>
16C9:0100  MOV  AH, 2     <ENTER>
16C9:0102  MOV  DL, 7     <ENTER>
16C9:0104  INT  21        <ENTER>
16C9:0106  MOV  AH, 4C    <ENTER>
16C9:0108  INT  21        <ENTER>
16C9:010A <ENTER>
-R CX <ENTER>
CX 0000
:A <ENTER>
-W <ENTER>
Writing 0000A bytes
-Q <ENTER>
```

By placing the BEEP command inside your DOS batch files, you can sound your computer's alarm to notify the user of key events, as in the following example:

```
@ECHO OFF
PAYROLL
BEEP
ECHO Payroll processing complete
ECHO Ready to print checks
PAUSE
PRTCHECK
```

DEBUG Script Files

To create the commands HELLO.COM and BEEP.COM, you invoked DEBUG and typed in the DEBUG commands and assembly language instructions. The most common way to create commands using DEBUG, however, is to create a file that contains the information you would typically type at DEBUG's prompt. These files, called *script files,* typically have the SCR extension. Once the file exists, you can use the DOS input redirection operator to create the command as shown here:

```
C:\> DEBUG  <  FILENAME.SCR  <ENTER>
```

The following script file, HELLO.SCR, contains the information needed to create the command HELLO.COM.

```
N HELLO.COM
A 100
MOV  AH, 9
MOV  DX, 10B
INT  21
MOV  AH, 4C
INT  21
DB   'Hello, world!$'
<ENTER>
R CX
19
W
Q
```

The only line that should be new is the line containing DEBUG's N command to name the file. Note the line containing only <ENTER>. As you recall, you must specify a blank line after the command's last assembly language instruction. The line containing only <ENTER> does just that.

Using HELLO.SCR, you can create HELLO.COM using DEBUG and the DOS input redirection operator as shown here:

```
C:\> DEBUG  <  HELLO.SCR
```

Using your word processor or the DOS editor, type in the following script file, HOUR.SCR, that creates the command HOUR.COM, which returns the current hour of the day as an exit status value.

```
N HOUR.COM
A 100
MOV AH, 2C
INT 21
MOV AL, CH
MOV AH, 4C
INT 21
<ENTER>
R CX
A
W
Q
```

You may recall from Chapter 16 that, using the IF ERRORLEVEL command, you can create a batch file named GREETING.BAT that displays a message based on the current time of day.

```
@ECHO OFF
HOUR
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 12 ECHO Good morning!
IF ERRORLEVEL 12 IF NOT ERRORLEVEL 18 ECHO Good afternoon!
IF ERRORLEVEL 18 ECHO Good evening!
```

As the complexity of your batch files increases, you will find that creating simple commands with DEBUG can be quite convenient.

Hands On

Using DEBUG, create the command Y_OR_N.COM that waits for the user to type either Y or N. If the user types Y, the batch file returns the exit status value 1. If the user types N, the batch file displays the exit status value 0.

```
C:\> DEBUG Y_OR_N.COM    <ENTER>
File not found

-A 100   <ENTER>
2707:0100 MOV AH, 0     <ENTER>
2707:0102 INT 16        <ENTER>
2707:0104 MOV AL, 0     <ENTER>
2707:0106 CMP AH, 15    <ENTER>
2707:0109 JE 112        <ENTER>
2707:010B CMP AH, 31    <ENTER>
2707:010E JE 114        <ENTER>
2707:0110 JMP 100       <ENTER>
2707:0112 MOV AL, 1     <ENTER>
2707:0114 MOV AH, 4C    <ENTER>
2707:0116 INT 21        <ENTER>
2707:0118  <ENTER>
-R CX   <ENTER>
CX 0000
:18   <ENTER>
-W   <ENTER>
Writing 00018 bytes
-Q   <ENTER>
```

Depending on your application, there may be times when you want AUTOEXEC.BAT to execute a specific command, and times you don't want the command executed. For example, you may or may not want AUTOEXEC.BAT to install memory-resident software, such as the DOS 5 or later DOSKEY command. Using Y_OR_N.COM, AUTOEXEC.BAT can prompt the user to determine whether to execute the command, as shown here:

```
ECHO Do you want to execute DOSKEY (Y/N)?
Y_OR_N
IF ERRORLEVEL 1 DOSKEY
```

Using your word processor or the DOS editor, create the file SCANCODE.SCR that returns the scan code corresponding to the key pressed by the user.

```
N SCANCODE.COM
A 100
INT 16
MOV AL, AH
MOV AH, 4C
INT 21
<ENTER>
R CX
8
W
Q
```

Using the DOS input redirection operator, create SCANCODE.COM as follows:

```
C:\> DEBUG  <  SCANCODE.SCR
```

Figure 29-1 displays the scancode values for each key. Using the scancode values, you can create the batch file MENU.BAT, which displays the following menu and waits for the user to press a key.

```
            M E N U


D  List directory files
C  Run CHKDSK
S  Sort directory listing
X  Exit to DOS
```

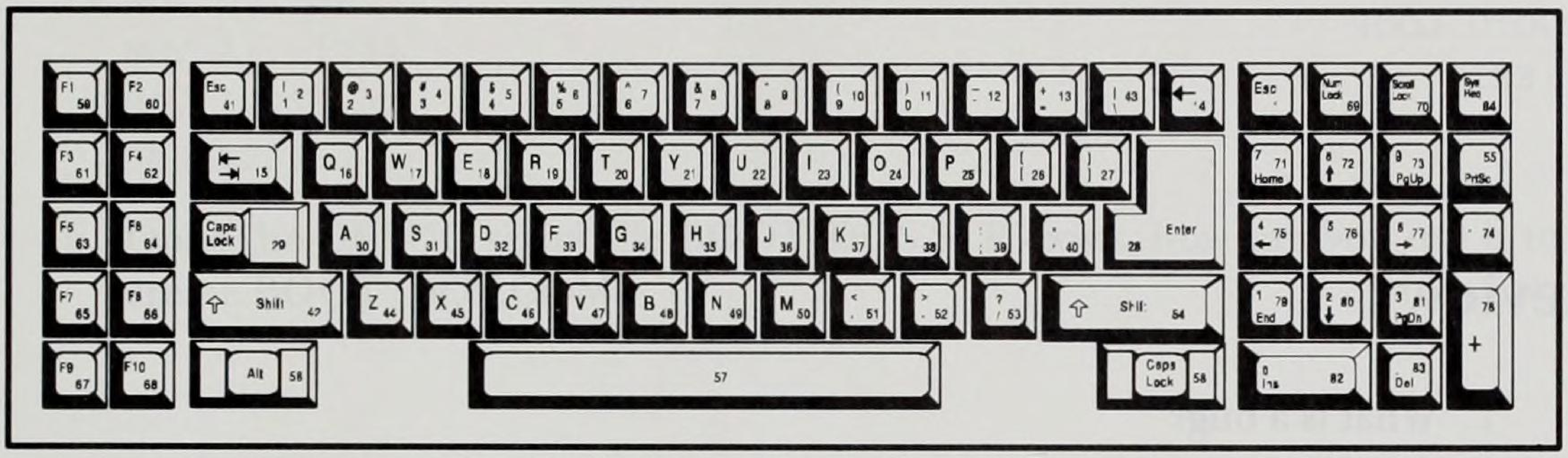


Figure 29-1. *Keyboard scancode values*

Using SCANCODE.COM, the batch file MENU.BAT determines the key pressed and proceeds accordingly.

```
@ECHO OFF
:LOOP
CLS
ECHO           M E N U
ECHO.
ECHO D  List directory files
ECHO C  Run CHKDSK
ECHO S  Sort directory listing
ECHO X  Exit to DOS
:GET_KEY
SCANCODE
IF ERRORLEVEL 32 IF NOT ERRORLEVEL 33 GOTO DIR
IF ERRORLEVEL 46 IF NOT ERRORLEVEL 47 GOTO CHKDSK
IF ERRORLEVEL 31 IF NOT ERRORLEVEL 32 GOTO SORT
IF ERRORLEVEL 45 IF NOT ERRORLEVEL 46 GOTO EXIT
GOTO GET_KEY
:DIR
DIR
PAUSE
GOTO LOOP
:CHKDSK
CHKDSK
PAUSE
GOTO LOOP
:SORT
```

```
DIR | SORT | FIND /V "e" | MORE
PAUSE
GOTO LOOP
:EXIT
```

Review

1. What is a bug?
2. How do you determine the size of the following command?

```
C:\> DEBUG GETDAY.COM <ENTER>
File not found

-A 100
16C9:0100  MOV AH, 2A <ENTER>
16C9:0102  INT 21     <ENTER>
16C9:0104  MOV AH, 4C <ENTER>
16C9:0106  INT 21     <ENTER>
16C9:0108
```

3. Using the following assembly language instructions, how would you use DEBUG to create the command SHOWLINE.COM?

```
MOV AH, 2
MOV SI, 82
MOV DL, [SI]
CMP DL, D
JZ 113
MOV AH, 2
INT 21
INC SI
JMP 105
MOV AH, 4C
INT 21
```

4. SHOWLINE.COM differs from ECHO because SHOWLINE does not write carriage return and linefeed characters, which advance the cursor to the next line. Change the batch file MENU.BAT (presented earlier in this chapter) to use SHOWLINE to prompt the user for an option.
5. Using the following assembly language instructions, create the DEBUG script file BOOT_DOS.SCR that reboots DOS.

```
MOV AX, 40
MOV DS, AX
MOV AX, 1234
MOV [72], AX
JMP FFFF:0
```

6. Once the DEBUG script file BOOT_DOS.SCR exists, how can you use it to create BOOT_DOS.COM?

Answers

1. A bug is an error in a software program.
2. Subtract the starting address from the ending address (108 – 100 = 8). Assign this value to the CX register and write the command as shown here:

```
-R CX
CX 0000
:8
-W
Writing 00008 bytes
-Q
```

GETDAY.COM returns, as an exit status value, the current day of the week. Sunday corresponds to the value 0, and Saturday corresponds to the value 6.

NOTE

3. Be sure to press ENTER at the end of each line shown in the following screen:

```
C:\> DEBUG SHOWLINE.COM        <ENTER>
File not found

-A 100  <ENTER>
2726:0100 MOV AH, 2            <ENTER>
2726:0102 MOV SI, 82           <ENTER>
2726:0105 MOV DL, [SI]         <ENTER>
2726:0107 CMP DL, D            <ENTER>
2726:010A JZ 113               <ENTER>
2726:010C MOV AH, 2            <ENTER>
2726:010E INT 21               <ENTER>
2726:0110 INC SI               <ENTER>
```

```
2726:0111 JMP 105      <ENTER>
2726:0113 MOV AH, 4C   <ENTER>
2726:0115 INT 21       <ENTER>
2726:0117   <ENTER>
-R CX       <ENTER>
CX 0000
:17         <ENTER>
-W          <ENTER>
Writing 00017 bytes
-Q          <ENTER>
```

4. The batch file would be changed as follows. Note the final ECHO command (with a period) to advance the cursor one line.

```
@ECHO OFF
:LOOP
CLS
ECHO       M E N U
ECHO.
ECHO D  List directory files
ECHO C  Run CHKDSK
ECHO S  Sort directory listing
ECHO X  Exit to DOS
SHOWLINE Enter choice:
:GET_KEY
SCANCODE
IF ERRORLEVEL 32 IF NOT ERRORLEVEL 33 GOTO DIR
IF ERRORLEVEL 46 IF NOT ERRORLEVEL 47 GOTO CHKDSK
IF ERRORLEVEL 31 IF NOT ERRORLEVEL 32 GOTO SORT
IF ERRORLEVEL 45 IF NOT ERRORLEVEL 46 GOTO EXIT
GOTO GET_KEY
:DIR
DIR
PAUSE
GOTO LOOP
:CHKDSK
CHKDSK
PAUSE
GOTO LOOP
:SORT
DIR | SORT | FIND /V "e" | MORE
```

```
PAUSE
GOTO LOOP
:EXIT
ECHO.
```

5. The script file would appear as follows.

```
N BOOT_DOS.COM
A 100
MOV AX, 40
MOV DS, AX
MOV AX, 1234
MOV [72], AX
JMP FFFF:0
<ENTER>
R CX
10
W
Q
```

6. Redirect the file BOOT_DOS.SCR into DEBUG using the DOS input redirection operator as shown here:

```
C:\> DEBUG < BOOT_DOS.SCR
```

Advanced Concepts

If you are an "old time" DOS user, you may use DEBUG extensively. Unfortunately, the size of this book prevents us from examining all of DEBUG's capabilities in detail. This section will briefly summarize each of DEBUG's commands. For the most part, DEBUG has remained unchanged over the past DOS versions. Table 29-1 lists the DEBUG commands.

The balance of this chapter summarizes each DEBUG command.

?

Function: Lists the available DEBUG commands.

Explanation: Displays a summary of the DEBUG commands and formats to your screen. This command requires DOS 5 or later.

Command	Function
?	Displays a list of available DEBUG commands (DOS 5 or later)
A	Lets you enter assembly language statements
C	Compares two memory areas
D	Displays a memory range
E	Enters data into a memory range
F	Fills a memory range with a specific value
G	Runs the command you are debugging from the current location to completion
H	Performs hexadecimal addition and subtraction
I	Displays the value in the port specified
L	Loads a file or disk sectors into memory
M	Copies a range of memory values from one location to another
N	Specifies the name of the file to be used with L and W commands
O	Writes a byte value to the port specified
P	Executes a loop, a repeated string instruction, a subroutine, or a software interrupt
Q	Quits a debugging session, returning control to DOS
R	Displays the contents of one or more registers, letting you change the register's value
S	Searches memory for the byte pattern specified
T	Executes the current instruction, displaying the register contents, the status flags, and the assembly language form of the instruction that DEBUG will execute next
U	Displays the assembly language representation for a range of memory values
W	Writes the current memory contents to the file specified by the N command
XA	Allocates a region of expanded memory (DOS 4 or later)
XD	Deallocates a region of expanded memory (DOS 4 or later)
XM	Maps expanded memory pages (DOS 4 or later)
XS	Displays the status of expanded memory (DOS 4 or later)

Table 29-1. The DEBUG Commands

Example: The following ? command displays the DEBUG commands:

```
-?
assemble     A [address]
compare      C range address
dump         D [range]
enter        E address [list]
fill         F range list
go           G [=address] [addresses]
hex          H value1 value2
input        I port
load         L [address] [drive] [firstsector] [number]
move         M range address
name         N [pathname] [arglist]
output       O port byte
proceed      P [=address] [number]
quit         Q
register     R [register]
search       S range list
trace        T [=address] [value]
unassemble   U [range]
write        W [address] [drive] [firstsector] [number]
allocate expanded memory        XA [#pages]
deallocate expanded memory      XD [handle]
map expanded memory pages       XM [Lpage] [Ppage] [handle]
display expanded memory status  XS
```

A [*memory address*]

Function: Lets you enter one or more assembly language instructions.

Explanation: Using DEBUG's A command you can change a command's instruction or, if you are creating a custom command, type in a series of instructions.

The memory address specifies the command's location. When you create your own commands, use A 100. The hexadecimal value 100 corresponds to the value 256. This tells DEBUG to enter the command's first instruction immediately after the 255-byte program segment prefix, which contains DOS-specific information for the command, such as its command line.

Example: The following example creates a command named PRTSC.COM that prints your screen contents as though you had pressed SHIFT-PRTSC.

```
C:\> DEBUG PRTSC.COM    <ENTER>
File not found

-A 100                  <ENTER>
2726:0100 INT 5         <ENTER>
2726:0102 MOV AH, 4C    <ENTER>
2726:0104 INT 21 <ENTER>
2726:0106
-R CX     <ENTER>
CX 0000
:6        <ENTER>
-W        <ENTER>
Writing 00006 bytes
-Q        <ENTER>
```

C *AddressRange StartingAddress*

Function: Compares the values stored in two or more memory locations, displaying differences.

Explanation: *AddressRange* specifies both a starting and ending address (or starting address and length) of the first memory range. *StartingAddress* specifies only the starting of the second range. If you don't include segment addresses, DEBUG uses DS. DEBUG compares the corresponding byte of the first range with each byte of the second range. If the two byte values differ, DEBUG displays the addresses and differing values in the following format.

`Address1 ByteValue1 ByteValue2 Address2`

If DEBUG finds no differences, DEBUG simply redisplays its prompt.

Examples: The following command compares 7 bytes beginning at 0100 to 0106 and 0200:

```
-C 0100 0106 0200 <ENTER>
```

Because the command does not specify segment addresses, DEBUG uses DS. The following command compares the first 255 bytes (FF in hexadecimal) of the data segment to the extra segment.

```
-C DS:0 LFF ES:0 <ENTER>
```

D [*Address* [*Number of Bytes*]]

Function: Displays the values currently stored in a range of memory.

Explanation: The D command directs DEBUG to display the values stored in a range of memory using the value's hexadecimal and ASCII representation. *Address* specifies the starting address of the first value to display. If you don't specify a segment address, DEBUG uses DS. If you don't specify the optional *NumberofBytes,* DEBUG displays 128.

DEBUG displays the memory dump as follows:

```
-D
2707:0100  59 58 86 C4 9E C3 8B 1E-02 96 4B 89 1E 02 96 A0   YX........K.....
2707:0110  97 03 0A C0 74 20 E8 06-00 B0 FF A2 34 00 F6 26   ....t ......4..&
2707:0120  F2 46 AC 8D 1E 08 96 FE-07 FE 0F 75 01 C3 FE 07   .F.........u....
2707:0130  E8 9A D5 75 01 C3 8B 1E-06 96 9F 4B 9E 89 1E 06   ...u.......K....
2707:0140  96 C3 8B 1E 04 96 9F 4B-9E 89 1E 04 96 E8 A1 FD   .......K........
2707:0150  E8 E4 BF 89 1E 06 96 C3-8B 1E 62 47 E8 BF BF 4A   ..........bG...J
2707:0160  8A E9 8A C7 0A C3 75 01-C3 43 8B F2 AC 8A C8 24   ......u..C.....$
2707:0170  7F 3C 0A 74 04 3C 0C 75-E6 8A C5 EB A6 9F 86 C4   .<.t.<.u........
```

If a byte value does not correspond to a printable ASCII character, DEBUG displays the value's ASCII representation as a period.

Examples: The following command displays the command's 255-byte program segment prefix:

```
-D 0 FF <ENTER>
```

Likewise, the following command displays the next 128 bytes of the data segment:

```
-D <ENTER>
```

E *StartingAddress* [*DataValues*]

Function: Enters byte values into successive memory locations.

Explanation: *StartingAddress* specifies the first address into which you want to assign one or more bytes or ASCII values. If you do not include a segment address, DEBUG uses DS. The optional *DataValues* is one or more hexadecimal byte values separated by spaces, tabs, or commas, or an ASCII string enclosed by single or double quotes.

If you don't specify *DataValues*, DEBUG will display each memory location's current value followed by a period, letting you type in the desired value. If you press ENTER after typing the value, DEBUG updates memory and the command ends. If you instead press the SPACEBAR, DEBUG will update memory and display the byte value stored in the next memory location, letting you continue the update. DEBUG lets you enter up to eight values on one line.

If you type in an invalid value, DEBUG will ignore it.

Examples: The following command assigns the ASCII string "DOS Book" to the memory location 100 relative to data segment:

```
-E  100  'DOS Book'
```

Likewise, the following command lets you enter byte values for a sequence of memory locations beginning at location 100:

```
-E  100
```

DEBUG will display location 100's current value followed by a period.

```
-E 100
2707:0100  44.
```

Type in the location's value and press ENTER to end the command or the SPACEBAR to enter a byte value in location 101.

F *AddressRange DataValues*

Function: Fills a range of memory with the value or values specified.

Explanation: *AddressRange* specifies the starting and ending address of the memory range to fill. The *DataValues* are one or more hexadecimal byte values separated by spaces, commas, or tabs or an ASCII character string enclosed by single or double quotes. If the number of data values is smaller than the range of memory, DEBUG repeatedly uses the values until the memory range is full.

Example: The following command fills a CGA screen with the word "DOS" using the ASCII values 44, 4F, and 53 for the letters D, O, and S and 20 for a space.

```
-F B800:0 4000 44,7,4F,7,53,7,20,7 <ENTER>
```

G [=*StartingAddress*] [*BreakPoint1*[*... BreakPoint10*]]

Function: Executes the current command either to completion or to an address specified.

Explanation: When you debug a program, you typically try to divide it into pieces you can individually verify as working or containing a bug. A *breakpoint* is a location within the program at which you want execution to stop, so you can examine either memory or the current register values. When you stop a program's execution using DEBUG, you need a way to get the program restarted. DEBUG's G (Go) command lets you do just that.

If you don't specify a breakpoint, DEBUG will run the program until it successfully completes, at which time DEBUG display's the message "Program terminated normally," or DEBUG will run the program until the program fails.

The =*StartingAddress* value specifies the address of the instruction at which you want DEBUG to begin the program's execution. If you don't specify a starting address, DEBUG uses the current value of the CS and IP registers. If you don't specify a breakpoint, DEBUG tries to run the program to completion. If you specify a breakpoint, DEBUG only stops the program's execution if the program attempts to execute the instruction stored at the breakpoint's address. DEBUG lets you specify up to ten breakpoints. If you don't specify a segment address, DEBUG uses CS.

Examples: The following command directs DEBUG to run the program from the current address in CS and IP to completion:

```
-G <ENTER>
```

This command restarts the program from address 100:

```
-G =100 <ENTER>
```

Last, this command resumes the program's execution while setting three breakpoints:

```
-G 200 300 400 <ENTER>
```

H *HexValue1 HexValue2*

Function: Displays the sum and difference of two hexadecimal numbers.

Explanation: Because DEBUG works in hexadecimal, determining the correct hexadecimal values to use for offsets or sizes can be difficult.

DEBUG's H command adds and subtracts two hexadecimal values, displaying first the sum, then the difference. DEBUG's H command is convenient for determining the size of simple commands you interactively create using DEBUG.

Example: The following command displays the sum and difference (in hexadecimal) of the values 9 and 8:

```
-H 9 8 <ENTER>
0011 0001
```

The hexadecimal value 11 corresponds to the decimal value 17.

I *PortAddress*

Function: Displays the byte value stored at the specified port address.

Explanation: Your computer exchanges information with hardware devices in one of two ways: First, the computer uses specific memory locations to address the hardware, such as memory-mapped video output. Second, your computer uses ports assigned to specific hardware devices, each individually addressed from 0 to FFFF. DEBUG's I command reads the value currently stored in the port.

Example: The following I command reads the port 378, which contains the last character sent to the parallel printer:

```
-I 378 <ENTER>
0A
```

In this case the 0A indicates that the last character sent to the printer was a linefeed character.

L [*StartingAddress*]

or

L *StartingAddress DriveNumber SectorNumber Sector Count*

Function: Loads either a file or specific disk sectors into memory.

Explanation: The first form of DEBUG's L command loads a file into memory at the address specified. DEBUG loads the file previously specified by an N (name) command. If you don't specify a segment address, DEBUG uses CS. If you don't specify an address at all, DEBUG loads files with the COM extension at location 100 and files with the EXE at the memory location specified in the program's header information.

The second form of DEBUG's L command loads one or more disk sectors. *StartingAddress* specifies the location at which DEBUG loads the first sector. If you don't specify a segment address, DEBUG uses CS. *DriveNumber* is a value from 0 to 25 where 0 corresponds to drive A and 25 corresponds to drive Z. *SectorNumber* specifies the first sector to load. *SectorCount* specifies the number of sectors.

Examples: The following commands select the file LABEL.COM and load the file into memory:

```
-N  LABEL.COM  <ENTER>
-L  <ENTER>
```

Likewise, this command loads the drive C's (drive number 2) boot record (sector number 0) at location 100 and then displays the sector:

```
-L  100  2  0  1  <ENTER>
-D  100  <ENTER>
```

M *AddressRange TargetAddress*

Function: Copies the values in one range of memory to another.

Explanation: *AddressRange* specifies the starting and ending memory addresses (or starting address and length) for the range of memory to copy. If you don't specify a segment address, DEBUG uses DS. *TargetAddress* is the address of the starting location to which you want the data copied. If the two memory ranges overlap, DEBUG will copy the data correctly.

Example: The following command copies 32 (hexadecimal 20) bytes of data beginning at location 0 to location 100:

```
-M 0 20 100 <ENTER>
```

N *Filename.Ext* [*CommandLine*]

Function: Specifies the name of a file to be loaded by an L command or to be written to by W.

Explanation: The filename can include a drive letter and directory path. When you issue the N command, DEBUG builds a 255-byte program segment prefix (PSP) at locations 0 to FF. The optional *CommandLine* is information DEBUG places at the hexadecimal offset 80 within the PSP. This information could possibly include a filename or other command information you want to use during debugging.

Examples: The following command selects the filename TEST.COM:

```
-N TEST.COM <ENTER>
```

This command selects TEST.COM and specifies command-line information that appears at DS:80:

```
-N TEST.COM Command line information <ENTER>
```

O *PortAddress ByteValue*

Function: Outputs a byte value to a hardware port.

Explanation: Your computer exchanges information with hardware devices in one of two ways: First, the computer uses specific memory locations to address the hardware, such as memory-mapped video output. Second, your computer uses ports assigned to specific hardware devices, each individually addressed from 0 to FFFF. DEBUG's O command writes a value to the port specified. Use the DEBUG O command with caution. If you write the wrong value to a port, or write to the wrong port, you may hang up your system.

Example: The following O command writes the value 61 to the port 378, which is the parallel port data register:

```
-O 378 61 <ENTER>
```

In this case the hexadecimal representation of a lowercase A is placed in the printer output register. Additional port output commands would be required to actually transfer the character from the adapter to the printer.

P [=*StartingAddress*][*InstructionCount*]

Function: Directs DEBUG to proceed through a loop, repeated string instruction, subroutine, or software interrupt.

Explanation: When searching for errors, programmers often use DEBUG to step through instructions one at a time. If you know that the next, possibly time-consuming, instruction is not the cause of the error, you can direct DEBUG to proceed through the instruction located at =*StartingAddress* in one step, or until the number of instructions specified by *InstructionCount* have executed.

If you don't specify a segment address for =*StartingAddress*, DEBUG uses CS. If you don't specify an instruction count, DEBUG uses the value 1. Using DEBUG's P command is very convenient for bypassing all of the instructions that correspond to INT 21.

Examples: The following command directs DEBUG to proceed through the current instruction:

```
-P  <ENTER>
```

The next command directs DEBUG to proceed through the software interrupt at address 112:

```
-P  =112  <ENTER>
```

Q

Function: Quits DEBUG, returning control to the DOS prompt.

Explanation: Q ends your debugging session. If you are using DEBUG to create simple commands, make sure you first write the command's instructions to a file on disk.

Example: The following command exits DEBUG to DOS:

```
-Q  <ENTER>
```

R [*RegisterName*]

Function: Displays the contents of one or more registers, and lets you make changes.

Explanation: *RegisterName* is the two-letter name of one of the 8086 registers or the single-letter Flags register.

AX BX CX DX SP BP SI DI DS CS SS ES IP PC
F

The Flags register F contains the current processor status codes. DEBUG displays the following status code symbols:

OV	Overflow flag set
NV	No overflow
UP	Direction up for string operations
DN	Direction down for string operations
EI	Interrupts enabled
DI	Interrupts disabled
NG	Negative sign
PL	Positive sign
ZR	Zero result
NZ	Nonzero result
AC	Auxiliary carry
NA	No auxiliary carry
PE	Even parity
PO	Odd parity
CY	Carry
NC	No carry

If you specify a *RegisterName,* DEBUG will display the register's current value and then prompt you (using a colon) to enter a new value. If you press ENTER at the colon prompt, DEBUG will leave the register's value unchanged.

If you invoke R with the Flags register F, DEBUG will display the status codes followed by a hyphen.

```
-R F   <ENTER>
NV UP EI PL NZ NA PO NC
-
```

To change one or more status codes, simply type the new setting or settings separated by spaces after the hyphen.

Examples: The following command displays the current register values:

```
-R  <ENTER>
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=2707  ES=2707  SS=2707  CS=2707  IP=0100   NV UP EI PL NZ NA PO NC
2707:0100 0000          ADD     [BX+SI],AL
DS:0000=CD
```

The following command assigns the CX register the value 5:

```
-R  CX  <ENTER>
CX  0000
:5  <ENTER>
```

The next command sets the overflow flag:

```
-R  F  <ENTER>
NV UP EI PL NZ NA PO NC
-OV  <ENTER>
```

S *AddressRange ByteValues*

Function: Searches memory for a byte pattern matching the values specified.

Explanation: *AddressRange* specifies either the starting and ending addresses for the range of memory DEBUG is to search, or the starting address and the number of bytes to search. To specify an address and a length, precede the second value with the letter L. For example, L10 directs DEBUG to search 16 bytes (10 is the hexadecimal representation for the value 16). If you don't specify a segment address for the starting address, DEBUG uses DS. If you don't specify a segment address for the ending address, DEBUG uses the same segment address as the first value.

The *ByteValues* are one or more byte values separated by spaces, commas, or tabs or a character string enclosed by single or double quotes.

Each time DEBUG finds a match to the pattern, DEBUG will display the segment and offset addresses of the first matching byte.

Examples: Give the following command to search the 64Kb data segment for the 'PATH=' string:

```
-S  O  FFFF  'PATH='  <ENTER>
```

Likewise, the next command searches the first 255 bytes of the data segment for the same string:

```
-S  O  LFF  'PATH='  <ENTER>
```

T [=*StartingAddress*] [*InstructionCount*]

Function: Executes the number of instructions specified, displaying the contents of the CPU registers and status flags after each instruction.

Explanation: To debug programs, programmers must often "single-step" through the program, executing an instruction and then examining registers and memory to verify the instruction was successful. This process of single-stepping through a program is called *tracing*. The DEBUG T command lets you trace one or more instructions.

The =*StartingAddress* value specifies the address of the instruction at which you want the program trace to begin. If you don't specify an address, DEBUG uses the address in CS and IP. *InstructionCount* specifies the number of instructions DEBUG should execute. If you don't specify an instruction count, DEBUG uses the value 1.

To suspend scrolling of a program trace, press CTRL-S; to cancel the program trace, use CTRL-C.

Examples: The following command traces the current instruction:

```
-T  <ENTER>
AX=0000  BX=0000  CX=20FA  DX=0000  SP=0080  BP=0000  SI=0000  DI=0000
DS=2726  ES=2726  SS=2971  CS=2930  IP=0012   NV UP EI PL NZ NA PO NC
2930:0012 8CC0          MOV     AX,ES
```

Likewise, this command traces five instructions:

```
-T 5 <ENTER>
```

Last, this command traces three instructions, beginning at location 100:

```
-T =100 3 <ENTER>
```

U [*AddressRange*]

Function: Displays the byte values in a range of memory in their assembly language representation.

Explanation: DEBUG's U, or unassemble, command displays a range of memory values as assembly language instructions. The U command lets you quickly examine a program's instructions, several instructions at a time. *AddressRange* specifies a starting and an ending address for the memory range to unassemble, or a starting address and byte length. If you don't specify a starting address, DEBUG uses the address in CS and IP. If you don't specify a segment address, DEBUG uses CS. If you don't specify an ending address or byte length, DEBUG unassembles 32 bytes of memory. To specify a byte length, place the letter L before the second value. For example, L10 directs DEBUG to unassemble 16 bytes (10 is the hexadecimal representation of 16).

Examples: The following command unassembles the range 100 to 110:

```
-U 100 110 <ENTER>
```

The following command unassembles eight bytes, beginning at DS:00:

```
-U 0 L8 <ENTER>
```

W [*StartingAddress*]
or
W *StartingAddress DriveNumber SectorNumber Sector Count*

Function: Writes the contents of a memory range to a file specified by a previous L command, or writes a range of memory directly to disk sectors.

Explanation: The most common use of DEBUG's W command is to write the instructions to a file of a simple command you have created with DEBUG. *StartingAddress* is the memory address of the first byte DEBUG is to write to disk. If you don't specify a starting address, DEBUG uses CS:0100.

DEBUG uses the register combination BX:CX to determine the number of bytes to write.

NOTE

Before you issue a W command, make sure BX and CX contain the correct length values. For small files (less than 64Kb), BX will be 0.

DEBUG's W command also lets you write directly to specific disk sectors. *DriveNumber* is a value from 0 to 25, where 0 corresponds to drive A and 25 corresponds to drive Z. *SectorNumber* is the first sector to be written, and *SectorCount* specifies the number of sectors to be written.

WARNING

The slightest mistake when writing to disk sectors may render your disk unusable, losing all the information the disk contains. If you must write to a disk sector, do so with extreme caution.

Example: The following command writes the memory range beginning at CS:0100 to the file previously named with a DEBUG N command. The number of bytes to write must be stored in BX:CX.

```
-W  <ENTER>
```

XA [*PageCount*]

Function: Allocates a specified number of expanded memory pages.

Explanation: *PageCount* specifies the number of 16Kb pages DEBUG is to allocate. If sufficient pages are available to meet the allocation request, DEBUG will display the hexadecimal value of the memory handle created. Otherwise, DEBUG will display an error message. See Chapter 32 for a discussion on expanded memory.

Example: The following command allocates 2 expanded memory pages:

```
-XA  2  <ENTER>
Handle created = 0004
```

XD [*Handle*]

Function: Deallocates the expanded memory pages that correspond to the memory handle specified.

Explanation: *Handle* is the hexadecimal value of the handle value displayed by DEBUG's XA command. If you specify an invalid handle number, DEBUG will display an error message. If the handle number is valid, DEBUG will display a message stating the handle (and its corresponding memory) was deallocated. See Chapter 32 for a discussion on expanded memory.

Example: The following command deallocates the memory corresponding to handle 0004:

```
-XD  0004  <ENTER>
Handle 00004 deallocated
```

XM *LogicalPage PhysicalPage Handle*

Function: Maps the specified logical page number to a physical page number.

Explanation: *LogicalPage* is a page number relative to the *Handle* specified. *Physical Page* is an actual page number within expanded memory. If the page numbers and handle specified are valid, DEBUG will display a message indicating the mapping was successful. Otherwise, DEBUG will display an error. See Chapter 32 for a discussion on expanded memory.

Example: The following command maps logical page 1 of handle 0004 to physical page 1:

```
-XM  1  1  0004  <ENTER>
Logical page 01 mapped to physical page 01
```

XS

Function: Displays the current status of expanded memory.

Explanation: XS displays the physical page addresses and the number of allocated pages. See Chapter 32 for a discussion on expanded memory.

Example: The following command displays the current expanded memory status:

```
-XS <ENTER>
```

Key Terms

Bug An error within a program that prevents the program from correctly executing.

Debug The process of removing errors from a program.

Debugger A software program that helps programmers debug their programs.

DEBUG script file A text file, typically with the SCR extension, containing the DEBUG commands and assembly language instructions needed to create a simple COM command.

Chapter 30

Another Look at Disks

Throughout the first 29 chapters of this book, you have made extensive use of floppy and hard disks. In this chapter you'll round out the remaining DOS disk commands: LABEL, VOL, DISKCOMP, SHARE, VERIFY, and RECOVER.

Disk Volume Labels: LABEL and VOL

A *disk volume label* is a name you assign to a disk. Each time you issue a DIR command, DIR displays your disk name, as in the following directory listing:

```
C:\> DIR <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 168C-5277
 Directory of C:\

DOS          <DIR>      03-16-93   4:44p
DISKUTIL     <DIR>      03-16-93   4:55p
    :          :           :         :
BACKUP       <DIR>      03-22-93   2:27p
CONFIG   SYS        256 02-20-93   9:55a
AUTOEXEC BAT        256 02-20-93   9:55a
       12 file(s)      412796 bytes
                     14820608 bytes free
```

If the disk does not have a label, DIR displays the following line instead:

```
Volume in drive C has no label
```

Users most commonly assign disk volume labels to floppy disks. Consider the following scenario. Each month you copy the contents of the four spreadsheet files BUDGET.DAT, EXPENSES.DAT, PAYROLL.DAT, and SALARY.DAT to floppy disk. A directory listing of the disk in drive A reveals the following files:

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 2C27-0ECF
 Directory of A:\

 BUDGET   DAT     40832 07-18-93  12:08p
 EXPENSES DAT     18944 07-18-93   3:27p
 PAYROLL  DAT     40576 07-16-93   2:22p
 SALARY   DAT     44800 07-21-93   8:23a
        4 file(s)     145152 bytes
                      216064 bytes free
```

In this case, the disk does not have a volume label. Assume that you give the disk to another user who consolidates data from several months to create year-to-date summaries. Before the user places your floppy disk into the drive, the user can look at the label attached to the disk to determine the month to which the files correspond. Once the user places the disk into the disk drive, the filenames for one month look just like the names for another. Should the user forget which month's disk is in drive A, the user must remove the disk and look at the label. Using a disk volume label, however, such as JUNE, JULY, or AUGUST, the user can quickly determine the file's month using the DIR command.

```
C:\> DIR  A: <ENTER>

 Volume in drive A is JULY
 Volume Serial Number is 2C27-0ECF
 Directory of A:\

BUDGET   DAT     40832 07-18-93  12:08p
EXPENSES DAT     18944 07-18-93   3:27p
PAYROLL  DAT     40576 07-16-93   2:22p
```

```
SALARY   DAT      44800 07-21-93   8:23a
        4 file(s)     145152 bytes
                      216064 bytes free
```

Assigning a Disk Volume Label

In Chapter 7 you learned that FORMAT lets you assign a disk volume label after it successfully formats a disk.

```
Format complete.

Volume label (11 characters, ENTER for none)?
```

If you have a formatted disk that contains information, you can use the LABEL command to assign a disk volume label. LABEL is an external DOS command that resides in the DOS directory. If you include a drive specifier in LABEL's command line, LABEL will assign the volume label to the disk specified. If you don't specify a disk, LABEL uses the default drive.

The following LABEL command assigns a volume label to the disk in drive A:

```
C:\> LABEL  A: <ENTER>
```

LABEL will display the disk's current volume label and possibly the disk's serial number depending on your DOS version. LABEL will then prompt you to type in the disk's new volume name.

```
C:\> LABEL  A: <ENTER>

Volume in drive A has no label
Volume Serial Number is 168C-5277
Volume label (11 characters, ENTER for none)?
```

Disk volume names can contain up to 11 characters. DOS lets you use in your disk volume labels the letters of the alphabet and numbers, as well as the following characters:

! @ # % _ - { } $ ’ ‘ ~

In DOS 4 or later, you can use spaces within the name.

This command assigns the disk label AUGUST:

```
C:\> LABEL  A: <ENTER>

Volume in drive A has no label
Volume Serial Number is 168C-5277
Volume label (11 characters, ENTER for none)? AUGUST <ENTER>
```

If your label contains invalid characters, LABEL will display an error message and prompt you to type in a correct volume name.

```
Invalid characters in volume label
Volume label (11 characters,       for none)? AUGUST <ENTER>
```

If you assign the wrong disk label, invoke LABEL again and retype the label name. If you want to delete the current disk label, press ENTER at LABEL's prompt for a new volume label. LABEL will ask you if you want to delete the current label.

```
Volume label (11 characters, ENTER for none)? <ENTER>
Delete current volume label (Y/N)?
```

To delete the label, type **Y** and press ENTER. If you type **N** and press ENTER, LABEL will leave the current disk label unchanged.

If you include the label name in the command line, LABEL will use the label specified without prompting you for one. The following command assigns the disk volume label OCTOBER to the disk in drive A:

```
C:\> LABEL  A:OCTOBER <ENTER>
```

Because the command line contains the disk label, LABEL assigns the disk label as desired and returns control to the DOS prompt.

Displaying a Disk Volume Label

As you just learned, the DIR command displays the disk volume label before each directory listing of files. Likewise, the CHKDSK command displays the disk label and the date and time you assigned the label.

```
C:\> CHKDSK <ENTER>

Volume DOS          created 04-21-1993 8:46a
Volume Serial Number is 168C-5277

 200065024 bytes total disk space
     81920 bytes in 3 hidden files
    462848 bytes in 108 directories
  44699648 bytes in 2034 user files
 154820608 bytes available on disk

      4096 bytes in each allocation unit
     48844 total allocation units on disk
     37798 available allocation units on disk

    655360 total bytes memory
    562688 bytes free
```

In addition, the internal DOS command VOL displays the disk volume label for the disk specified. If you don't include a disk drive specifier in VOL's command line, VOL displays the volume label for the current disk.

```
C:\> VOL <ENTER>

 Volume in drive C is DOS
 Volume Serial Number is 168C-5277
```

If the disk does not have a label, VOL displays the following:

```
Volume in drive A has no label
 Volume Serial Number is 168C-5277
```

This command uses VOL to display the volume label for the disk in drive A:

```
C:\> VOL  A: <ENTER>
```

Comparing Two Floppy Disks with DISKCOMP

For many years after DOS was released in 1981, most users had floppy disk-based systems. It was common for users to copy the contents of the floppy disk in drive A to the floppy disk in drive B using DISKCOPY, as discussed in Chapter 12.

After completing a DISKCOPY operation, users would invoke the DISKCOMP command to ensure the DISKCOPY operation was successful. The DISKCOMP command compares two floppy disks track by track, sector by sector, and byte by byte, displaying any differences it encounters in the following format:

```
Compare error on
side n, track n
```

If the two disks contain the same contents DISKCOMP displays the following:

```
Compare OK
```

The following command compares the floppy disk in drive A to the floppy disk in drive B:

```
C:\> DISKCOMP  A:  B: <ENTER>
```

DISKCOMP will prompt you to place the first disk in drive A and the second disk in drive B.

```
Insert FIRST diskette in drive A:

Insert SECOND diskette in drive B:

Press any key to continue...
```

When you press any key, DISKCOMP will perform the disk comparison.

When the DISKCOMP operation completes, it will ask you if you want to compare additional disks.

```
Compare another diskette (Y/N) ?
```

If you type Y and press ENTER, DISKCOMP will prompt you to place disks into drives A and B, repeating the comparison process. If you type N, DISKCOMP will end, returning control to the DOS prompt.

DISKCOMP only compares floppy disks of the same size and capacity. You cannot use DISKCOMP to compare a 5 1/4-inch floppy to a 3 1/2-inch floppy.

If your computer only has a single floppy drive, simply specify the same drive letter as the second drive in DISKCOMP's command line, like this:

```
C:\> DISKCOMP A: A: <ENTER>
```

DISKCOMP will prompt you to insert either the first or second disk as required.

DISKCOMP only supports floppy disk drives. If you reference drive C, or only specify one floppy disk letter in the command line, DISKCOMP will display the following error message:

```
C:\> DISKCOMP <ENTER>

Invalid drive specification
Specified drive does not exist
or is non-removable
```

File Sharing and Large Disk Support with SHARE

If your computer is connected to other computers over a local area network, there may be times when two or more users try to use the same file at the same time. If you aren't connected to a local area network, but instead use a program such as Windows or the DOS 5 or later shell (using the task swapper) to run several programs at the same time, two of the programs may simultaneously access the same file.

The SHARE command, introduced with DOS 3, loads memory-resident software that manages file sharing. Assume, for example, the program is reading the contents of a file, while another program is trying to update the file. Depending on the timing of the two operations, the program reading the file could get part of the old information and part of the new.

To prevent such conflicts, SHARE lets a program lock a file or part of the file for exclusive use. When the program is finished with its operation, the program unlocks the file so another program can use it. The format of the SHARE command is as follows:

SHARE /F: *NumberofBytes* /L: *NumberofLocks*

When a program locks a file, SHARE records information about the file such as its path name and the region locked to a table in memory. Each locked file requires 11 bytes plus the length of the file's path name, up to 64 bytes. SHARE's /F switch lets you specify the number of bytes SHARE should allocate for the table. For most users, the default size of 2048 bytes is large enough.

SHARE's /L switch lets you specify the number of locks required. SHARE's default value is 20.

If your computer is connected to a local area network, your network administrator can tell you if you need to invoke SHARE. If you don't need to use SHARE, don't use it. SHARE's memory-resident software examines every file's read or write operation to prevent sharing conflicts. This examination requires processing time and, thus, slows down your programs.

One of the most common uses of SHARE is to correct a situation that arises in DOS 4 when you are using disk partitions larger than 32Mb.

DOS 4 was the first major DOS version to support disk partitions larger than 32Mb. When users with disk partitions larger than 32Mb started their systems in DOS 4, their screens would display a warning message about loading SHARE for large media.

Many users ignored the warning message and did not invoke the SHARE command. In fact, many users never experienced any adverse effects. These users were the lucky ones that never ran an older DOS program that accessed files using file control blocks (FCBs), discussed in Chapter 24.

Before version 2.0, DOS used file control blocks to access files. Older programs that use FCBs to access files, therefore, don't know about DOS 4's support of partitions larger than 32Mb. When one of these older programs ran on a large partition, the program might read and write disk information to the wrong locations. One disk write could destroy a file, directory, or even the file allocation table that tells DOS where each of your files reside on disk.

The DOS 4 SHARE command solves this FCBs problem. Because it is almost impossible to tell which programs use file control blocks, DOS 4 users with large disk partitions that don't invoke SHARE are playing "Russian Roulette" with their files.

If you are using DOS 4 and are using SHARE, place the following INSTALL entry in your CONFIG.SYS file to load the SHARE command:

```
INSTALL=C:\DOS\SHARE.EXE
```

DOS 5 or later corrects this problem and does not require you to install SHARE.

Verifying Disk Writes with VERIFY

When you copy a file to disk, or a program stores information to disk, DOS transfers the data to disk a sector at a time. In general, DOS directs the disk controller (hardware) to store the information in a specific sector location. The disk controller then actually oversees the information recording. By default, DOS assumes the disk controller correctly records the data on disk. In rare instances, however, the disk, due to a damaged surface area or hardware malfunction, may not record the data correctly. However, such disk storage errors are rare.

The VERIFY command lets you enable *disk verification,* which checks the information stored on disk to make sure the information was stored correctly. To enable disk verification, issue the command VERIFY ON as shown here:

```
C:\> VERIFY  ON <ENTER>
```

You can think of disk verification as the reading of the information written to disk followed by a comparison to the original data. If the data recorded on disk matches the original, the write operation was correct. If the data on disk does not match the original, DOS knows an error occurred. In most cases, the disk controller, not DOS, performs the actual data verification.

The additional processing required to perform disk verification will decrease your system performance. Because disk recording errors are rare, most users should only turn disk verification on before key file copy or backup operations.

To turn disk verification OFF, invoke VERIFY as follows:

```
C:\> VERIFY  OFF <ENTER>
```

If you invoke VERIFY without stating on or off, VERIFY will display its current state.

```
C:\> VERIFY <ENTER>
VERIFY is off
```

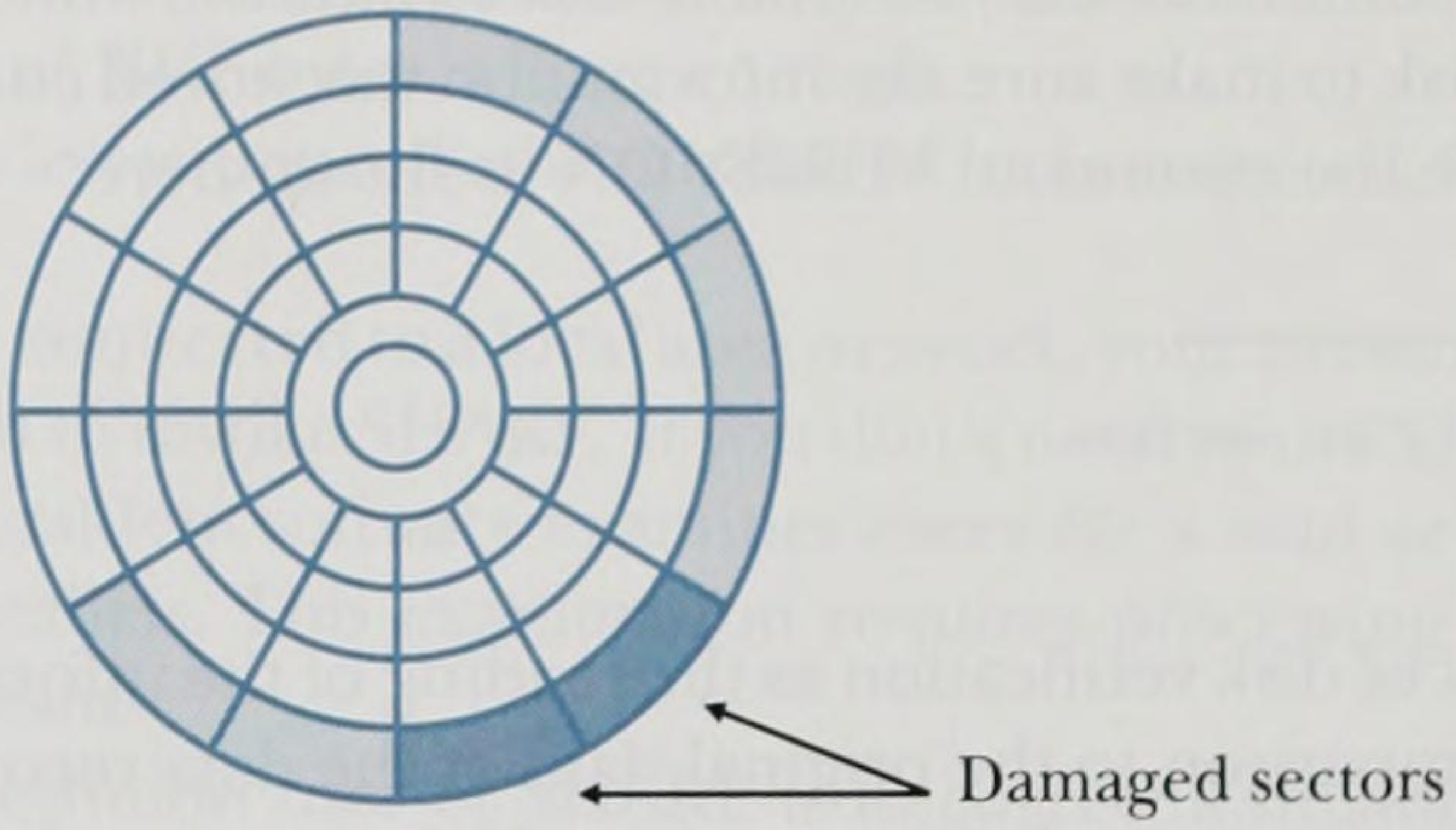
NOTE

Disk verification is useful when you copy key files or when you perform disk backup operations. To eliminate the need to explicitly turn disk verification on and off each time you copy key files, both COPY and XCOPY provide a /V switch, which turns disk verification on for only the duration of the copy operation. For more information on /V, see COPY and XCOPY in the Command Reference at the end of this book.

Salvaging a File or Disk with RECOVER

DOS stores your file contents on disk in sectors. Although such instances are rare, it is possible for a disk sector to become damaged, preventing your access to the file's information. When such an error occurs, DOS will display a disk read error message on screen, normally followed by the prompt "Abort, Retry, Fail?"

The RECOVER command lets you restore a portion of a file's contents, up to the damaged sector. DOS 6 does not provide the RECOVER command. For example, the following illustration shows a file containing damaged sectors:



Using RECOVER, you can copy the file's contents up to the point of the damaged sector. To salvage a file using RECOVER, invoke RECOVER with the filename, as shown here:

```
C:\> RECOVER  FILENAME.EXT  <ENTER>
```

RECOVER will rebuild the file up to the point of the bad sector; information beyond the bad sector will be lost.

In some cases, a damaged sector may prevent DOS from accessing the entire disk. Should such an error occur, invoke the RECOVER command with the disk drive letter, as shown here:

```
C:\> RECOVER  A: <ENTER>
```

RECOVER will examine the disk and attempt to salvage as many files as it can. RECOVER will place the files it salvages in the disk's root directory, naming them FILE0000.REC, FILE0001.REC, and so on. A directory listing of a recovered disk might appear as follows:

```
C:\> DIR  A: <ENTER>

 Volume in drive A has no label
 Volume Serial Number is 374A-0FF8
 Directory of A:\

FILE0001 REC      1024 04-21-93  10:19a
FILE0002 REC     35840 04-21-93  10:19a
```

```
FILE0003 REC     30720 04-21-93  10:19a
FILE0004 REC     46080 04-21-93  10:19a
FILE0005 REC      1024 04-21-93  10:19a
FILE0006 REC      7168 04-21-93  10:19a
FILE0007 REC     40960 04-21-93  10:19a
FILE0008 REC     26624 04-21-93  10:19a
        8 file(s)     189440 bytes
                      173056 bytes free
```

Use RECOVER as a last resort to salvage a file or disk. Never invoke RECOVER with a disk or file that is not damaged; doing so may destroy the information the file or disk contains. Do not rely on RECOVER. Instead, make sure you have current backup files. RECOVER does not undelete files. DOS 6 does not support RECOVER.

NOTE

SUMMARY

LABEL

The LABEL command assigns an 11-character disk volume label to the disk drive specified. The volume label is a name that helps you tell one disk from another. The following LABEL command assigns the disk volume label DOS to drive C:

```
C:\> LABEL  C:DOS <ENTER>
```

VOL

The VOL command displays a disk's 11-character volume name. If you don't specify a disk drive, VOL displays the volume label for the current disk. The following VOL command displays the volume label for the current disk:

```
C:\> VOL <ENTER>
```

DISKCOMP

DISKCOMP compares two floppy disks, track by track, sector by sector, and byte by byte. If the disks differ, DISKCOMP will display the side and track numbers that differ. If the disks are the same, DISKCOMP will display the

SUMMARY

"Compare OK" message. The following DISKCOMP command compares the floppy disk in drive A to the disk in drive B:

```
C:\> DISKCOMP  A:  B: <ENTER>
```

SHARE

SHARE loads memory-resident software that provides file-sharing support for local area networks or users running several programs that may share the same files at the same time.

In DOS 4, the SHARE command provides file control block support for disk partitions larger than 32Mb. If you are using DOS 4 and disk partitions greater than 32Mb, invoke SHARE using the CONFIG.SYS INSTALL entry as follows:

```
INSTALL=C:\DOS\SHARE.EXE
```

VERIFY

VERIFY lets you turn disk verification on and off. In rare instances, the information DOS tells the disk controller to store on disk does not get recorded correctly. In most cases, the recording error is due to a hardware malfunction. When disk verification is on, every sector written to disk is compared to the original data to ensure the data match. Although this extra checking ensures the integrity of your data, it requires additional processing, which decreases your system performance.

RECOVER

RECOVER lets you salvage a portion of a damaged file or disk. You should use RECOVER only as a last resort. If you invoke RECOVER with a file or disk that is not damaged, RECOVER may destroy the information the disk or file contains. DOS 6 does not support the RECOVER command.

Hands On

Place a formatted floppy disk in drive A and issue the following LABEL command:

```
C:\> LABEL <ENTER>
```

When LABEL prompts you for a volume label name, type **CHAPTER-30**.

Use the following DIR command to display the label name:

```
C:\> DIR  A: <ENTER>
```

Next, use VOL to display the label as follows:

```
C:\> VOL  A: <ENTER>
```

Depending on your DOS version, CHKDSK may display the disk label, as well as the date and time the label was assigned.

```
C:\> CHKDSK  A: <ENTER>
```

Next, invoke LABEL as follows to change the disk's label to DISKINFO:

```
C:\> LABEL  A:DISKINFO <ENTER>
```

Using the following VERIFY command, display the current state of disk verification as follows:

```
C:\> VERIFY
```

By default, disk verification should be off. If your system has disk verification on, check your AUTOEXEC.BAT for the VERIFY command. Because disk recording errors are so rare, enabling disk verification for all operations only reduces your system performance.

Last, if you are using DOS 4 and disk partitions larger than 32Mb, make sure you use the CONFIG.SYS INSTALL entry to invoke the SHARE command.

Review

1. What is a disk volume label?
2. List three commands that display the disk volume label.
3. What is the difference between DISKCOPY and DISKCOMP?
4. Why does DOS 4 require the SHARE command?
5. What is disk verification?

Answers

1. A disk volume label is an 11-character disk name. The FORMAT and LABEL commands let you assign a volume label to a disk.
2. DIR, CHKDSK, LABEL, and VOL all display the disk volume label.
3. DISKCOPY copies the contents of one floppy disk to another. DISKCOMP compares the contents of one floppy disk to another.
4. DOS 4 was the first major DOS version to support disk partitions larger than 32Mb. Unfortunately, many very old programs that used file control blocks to access files were not compatible with the larger disks. When such a program ran, the program could incorrectly write information to any disk location, overwriting existing files, directories, or even the file allocation table.
5. Although it is rare, it is possible for the disk controller to incorrectly record information on disk due to a damaged disk surface area or a hardware malfunction. Disk verification is the process of comparing the information recorded on disk to the original data. If the information matches, the disk operation was successful. If not, the disk controller can repeat the operation, recording the data correctly. VERIFY lets you enable and disable disk verification. Because such disk recording errors are rare, most users leave disk verification disabled to reduce system overhead. As a rule, you only need to enable disk verification before backup or key file copy operations.

Advanced Concepts

Although the entry does not actually show up in directory listings or output of ATTRIB, TREE, or other DOS commands, DOS actually stores the disk volume label in a root directory entry. As you will learn in Chapter 33, every directory entry contains 32 bytes of information. In the case of a disk volume label, the filename and extension fields provide storage for the label's 11-character name, and the date and time fields record when you assigned the label.

In DOS 5 or later, DOS also records the volume label in the disk's boot record.

Assuming you assign the label name DOS to your hard disk, the directory entry storing the volume name will be similar to the label shown here:

DOS			12-31-93	12:44 PM	Volume
Name	Extension	Reserved	Date	Time	Attributes

A Closer Look at File Control Blocks

In Chapter 24 you learned that earlier (before DOS 2) versions of DOS use file control blocks (FCBs) to perform file input and output operations. In this chapter you learned that, if you are using partitions in DOS 4 larger than 32Mb and have not invoked SHARE, programs using FCBs can destroy the information on your hard disk. Beginning with DOS 2, DOS began using file handles to track open files instead of file control blocks. Figure 30-1 illustrates a file control block.

As you can see, the file control block contains information about the file such as its name, size, date and time stamps, as well as record and block numbers that DOS uses to keep track of its current position in the file. Note that the file control block does not contain directory path information. DOS versions before 2.0 did not provide directory support. When version 2.0 introduced directories, DOS had to maintain its current support for file control blocks, while providing a way to access files in directories. The solution came in the form of file handles, used by all DOS programs today. A file handle is an index into a table of values DOS maintains about the currently open files.

Unfortunately, until users throw away all the older programs based on FCBs, file control blocks will continue to be a nuisance.

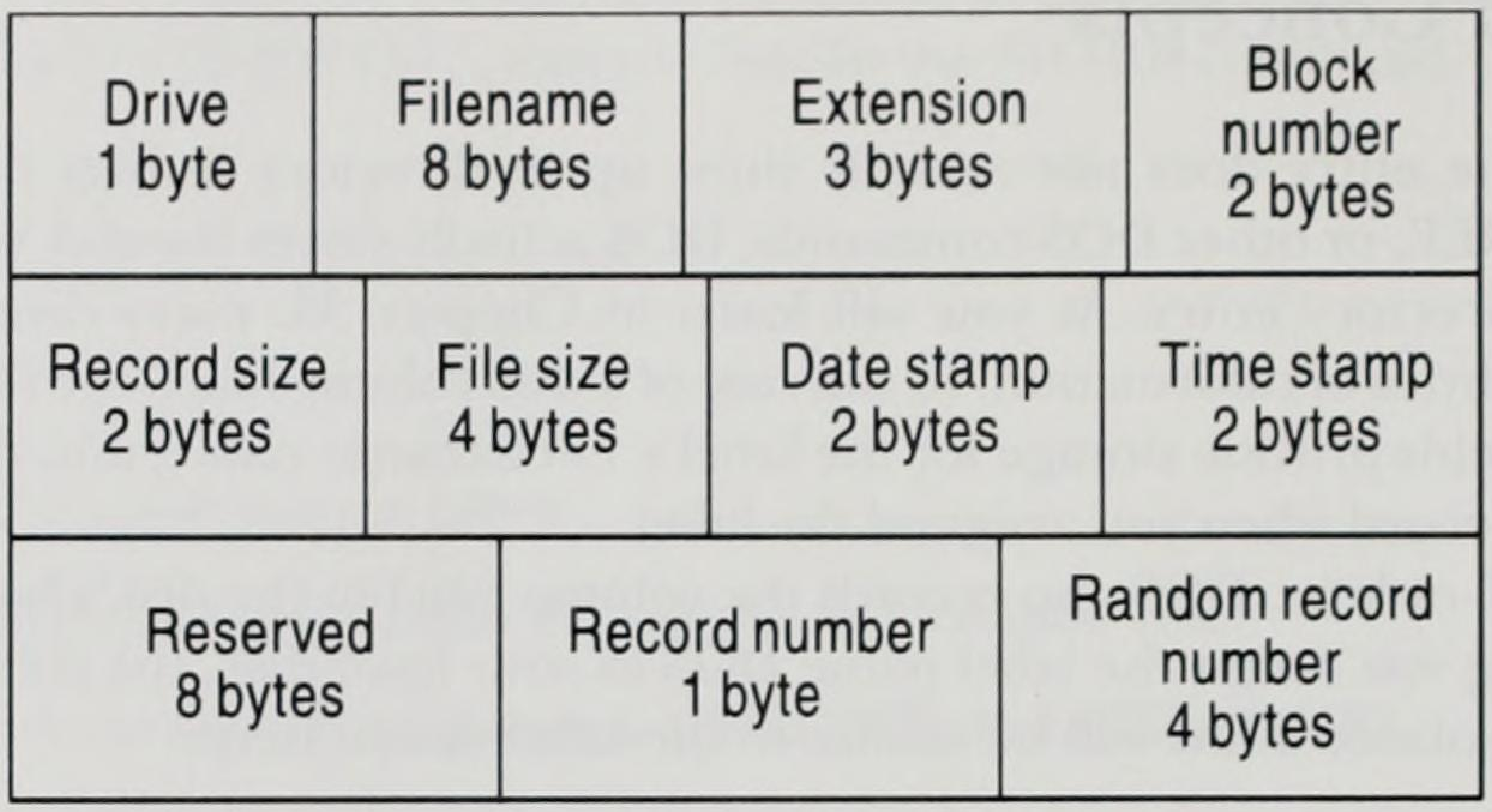

Drive
1 byte
Filename
8 bytes
Extension
3 bytes
Block number
2 bytes
Record size
2 bytes
File size
4 bytes
Date stamp
2 bytes
Time stamp
2 bytes
Reserved
8 bytes
Record number
1 byte
Random record number
4 bytes

Figure 30-1. *A file control block*

Key Terms

Volume label An 11-character name assigned to a disk.

Disk verification The process of comparing the data written to disk to the original information to verify no recording errors occurred.

Chapter 31

Improving Your System Performance

It doesn't take most users very long before they want their system to run faster. Unfortunately, it's difficult to trade in your existing computer each time a faster, more powerful machine comes along. What you can do, however, is make sure your system is configured for maximum performance. This chapter examines DOS commands and concepts that relate strictly to improving your computer's performance. In most cases, the discussion will be brief, giving you a reference to another chapter that describes the concept in detail. Your computer represents a large investment of your time and money. This chapter will help you get the most results from that investment.

Directory Considerations for Maximum Performance

Each time you run a program or use a file, DOS must search your disk to first locate the directory containing the file or program. Then DOS must read the directory from disk into memory to search for the file's corresponding directory entry. Using the directory entry, DOS can determine where the file resides on disk and can then read the file from disk as required. Depending on your directory structure, the number of files in each directory, and your command path (see PATH) or data-file search path (see APPEND), the amount of time DOS spends searching your disk for a program or file can be considerable. If you consider the number of times throughout the day you run programs or open files, the overhead resulting from repeated directory searches can become quite time consuming. Here are three steps you can take to reduce directory search overhead.

First, if you are using DOS 3.3 or later, the FASTOPEN command directs DOS to remember the starting location on disk of your recently used files. Assume, for example, you repeatedly edit the file BUDGET.RPT throughout the day. Each time you open the file, DOS must search the directory list to find the file's entry, so DOS can determine where the file starts on disk. To search the directory, DOS must locate the directory on disk, read the directory from disk into memory, and then search each entry until the file is found. FASTOPEN eliminates this directory searching overhead by placing the file's name and starting location in a table in your computer's fast electronic memory. When you open a file or execute an external command, DOS first checks FASTOPEN's table of entries to see if it can locate the file without having to perform the slow directory search. If the file is not in the table, DOS searches for the file on disk. After DOS locates the file, it places its name and location in FASTOPEN's table to be referenced when you access the file after that.

FASTOPEN will not decrease the time required to open every file on your disk, just those files you use more than once. Most users will find that the following FASTOPEN command improves their system performance:

```
C:\> FASTOPEN  C:=50 <ENTER>
```

In this case, FASTOPEN will track your last 50 open files. FASTOPEN lets you track up to 999 files. Using a large value such as 999, however, will consume a considerable amount of your computer's memory. Remember, the goal of using FASTOPEN is not to track every file you open on your disk, but rather, the files you use on a repeated basis. If you are using DOS 4, use the CONFIG.SYS INSTALL entry to load FASTOPEN's memory-resident software. If you are using DOS 5 or later, use the LOADHIGH command to load FASTOPEN into reserved memory.

One reason directory searches consume so much time is the number of files in the directory. DOS directories, discussed in Chapter 10, help you organize your files. To use directories most effectively, you need to keep the number of files in any directory to a manageable number. If you have installed a third-party program such as a word processor or spreadsheet, you may not have control over the number of files in the directory. However, if you are creating directories that contain your own files, you can control the number of files in the directory by further organizing the files into subdirectories.

To determine the maximum number of files to which you should ideally restrict your directories, you need to know your disk's cluster size. DOS allocates disk space to files and directories using consecutive groups of sectors called clusters or allocation units. If you are using DOS 4 or later, the CHKDSK command displays your disk's allocation unit size. For hard disks, the allocation units are typically 2048 or 4096 bytes. Each directory entry consumes 32 bytes. For best performance, you should restrict the number of files in a directory to the number of entries that will fit in a disk cluster.

For a cluster size of 2048, you should not exceed 64 files (2048 ÷ 32 = 64). Likewise, for 4096-byte clusters, restrict your directories to 128 files. If you place more files into the directory than will fit in one cluster, DOS must allocate a second cluster for the directory. This second directory cluster will very likely be fragmented (see Chapter 42), decreasing your system performance with every directory search operation.

As you have learned, the PATH command makes it very easy for you to execute external commands by letting you specify directories within which DOS will automatically search for commonly used commands. As a rule, you should only place the names of directories that are likely to contain commonly used commands in your command path. Otherwise, DOS may spend considerable time searching directories that are unlikely to contain the current command. In addition, put the directories containing your most commonly used commands (such as the DOS directory) at the beginning of the command path. Also, keep in mind that if the directory in your command path is fragmented because it contains a large number of files, your system performance will decrease every time DOS searches the directory for a command. If you are using APPEND to define a data-file search path, the same concepts apply.

The more files a directory contains, the longer it will take DOS to search the directory's contents. Depending on your DOS version, your DOS subdirectory may contain several commands you may never use. For example, the following files are used only by international DOS users:

COUNTRY.SYS
NLSFUNC.EXE
4201.CPI
LCD.CPI
KEYB.COM
DISPLAY.SYS
4208.CPI
PRINTER.SYS
KEYBOARD.SYS
EGA.CPI
5202.CPI

If you don't need to use the international commands, you can delete these files or move them into a directory below DOS. In addition, the DOS directory may contain several files with the BAS extension, which contains samples of BASIC programs. If you don't plan on using BASIC, you can delete those files or move them to a lower-level directory. If you take time to better organize the directories in your command path in this way, you will improve your system performance.

If you are not sure what function a command performs, or the contents of a specific file, do not move or delete the file.

File Considerations for Maximum Performance

In general, the biggest single factor that affects the performance of file-manipulation commands is fragmentation. A fragmented file is a file whose contents are spread out across your disk. Fragmentation occurs as a natural result of creating, deleting, and enlarging the files on your disk. If your programs seem to start slowly, or your data files in a word processing document or spreadsheet load slowly, your files may be fragmented. Chapter 44 discusses ways to identify and correct fragmentation.

When you use the DOS pipe operator (discussed in Chapter 21) to redirect one command's output to become a second command's input, DOS creates temporary files in your disk's root directory. If you are using DOS 5 or later, you can use the TEMP environment variable to direct DOS to create the files in a different location, ideally on a fast RAM disk. Chapter 32 discusses RAM disks in detail and describes how to create a RAM disk using the RAMDRIVE.SYS device driver. If your computer has extended or expanded memory, use the memory to create a RAM disk and then use the SET command to assign TEMP to point to the RAM disk. If your computer doesn't have extended or expanded memory, creating a RAM disk in this way may consume memory your programs could put to better use.

Disk Considerations for Maximum Performance

Chapter 35 discusses disk interleaving, a process that changes the order of sectors stored on your disk to improve the timing coordination between your disk controller and computer. If your computer is not fast enough to keep up with your disk, your system performance will suffer because your disk will repeatedly have to wait for disk sectors to complete an extra revolution before the computer is ready to receive them. This extra rotational delay can have a big performance impact. Unfortunately, to correct the disk's interleave, you will need to purchase a third-party disk utility. DOS does not provide the capability to determine and set the correct interleave value for your computer and disk.

Because disks are mechanical, they are much slower than the computer's electronic components. One of your goals, therefore, is to reduce disk operations whenever possible. In Chapter 24, you learned that the CONFIG.SYS BUFFERS entry lets you specify the number of buffers DOS should place in memory to reduce disk input and output operations. By increasing the number of available disk buffers, you can reduce the number of disk read operations DOS must perform. Each time an application asks DOS to read information from disk, DOS first checks its disk buffers to see if the information has already been read. If so, DOS can provide information to the program without having to perform a slow disk read operation. For more information on the correct number of disk buffers you should use, see Chapter 24.

Because DOS stores files on disk in groups of sectors called clusters, it is very likely that if a program asks DOS to read a file's first sector, the program will soon ask DOS to read the sectors that immediately follow. Beginning with DOS 4, the CONFIG.SYS BUFFERS entry lets you specify the number of sectors DOS can read in advance into a memory buffer. Should a program later request the sectors, DOS already has the sectors in memory, eliminating the need to perform another disk read operation. If you are using DOS 4 or later and you are not using the SMARTDRV disk-caching software, use the BUFFERS entry to enable disk read-ahead operations.

There are rare instances when the disk does not correctly record information. Using the VERIFY command (discussed in Chapter 30), you can enable disk verification, which instructs the disk to check to make sure the information was recorded correctly. Because such disk errors are rare, leaving disk verification enabled results in unnecessary checking on every disk write, thus decreasing your overall system performance. Using the following VERIFY command, you can determine if disk verification is enabled (on) or disabled (off):

```
C:\> VERIFY <ENTER>
VERIFY is off
```

If your system reports that verification is ON, check your AUTOEXEC.BAT file for a VERIFY ON command, and if such a command exists, remove it.

If you need to perform a file copy for critical data, the COPY and XCOPY commands both support the /V switch, which enables disk verification for the duration of the command. If you enable disk verification before your backup operations, remember to disable it when the backups are completed.

Creating a Disk Cache

In Chapter 24 you learned how disk buffers improve your system performance by reducing the number of disk I/O operations DOS must perform. As you also learned in Chapter 24, each disk buffer consumes 528 bytes of conventional memory. Depending on your DOS version, or if you are using Microsoft Windows, the SMARTDRV.SYS device driver or the command SMARTDRV.EXE lets you install disk-caching software into extended or expanded memory.

A *disk cache* is a very large disk buffer. Each time DOS needs to perform a disk read, DOS just searches the disk cache for the data. Disk caches have several advantages over DOS disk buffers. First, because disk caches reside in extended or expanded memory, they don't use conventional memory that DOS can use to hold program instructions. Second, disk caches can be very large, up to 8Mb. Last, most disk-caching software is very specialized and executes very fast.

To determine whether you should use SMARTDRV.SYS or SMARTDRV.EXE, issue the following command:

```
C:\> DIR \DOS\SMARTDRV.* <ENTER>
```

If DIR displays the filename SMARTDRV.SYS, follow the instructions listed next for installing the SMARTDRV device driver. If, instead, DIR displays the filename SMARTDRV.EXE, follow the instructions listed for invoking the SMARTDRV command.

Installing the SMARTDRV.SYS Device Driver

To install a disk cache, you must use the CONFIG.SYS DEVICE entry to load SMARTDRV.SYS. The format of SMARTDRV.SYS is

```
DEVICE=C:\DOS\SMARTDRV.SYS  [InitialCacheSize]
  [MinimumCacheSize]  [/A]
```

InitialCacheSize specifies the starting cache size in kilobytes, from 128Kb to 8192Kb (8 megabytes). The default cache size is 256Kb. Depending on your memory and cache usage, it is possible for some programs to reduce the cache size. *MinimumCacheSize* specifies the smallest size in Kb a program can reduce the cache. For example, using Microsoft Windows, you can use SMARTDRV.SYS to create a very large disk cache. Should Windows later need to put unused portions of the cache to use, Windows will reduce the cache size. The default value is 0. By default, SMARTDRV.SYS installs the cache in extended memory. The /A switch directs SMARTDRV.SYS to use expanded memory instead.

The following CONFIG.SYS entry installs a 256Kb disk cache in extended memory:

```
DEVICE=C:\DOS\SMARTDRV.SYS
```

When you start your computer, your screen will display a message similar to the following:

```
Microsoft SMARTDrive Disk Cache version 3.13
    Cache size: 256K in Extended Memory
    Room for 16 tracks of 32 sectors each
    Minimum cache size will be 0K
```

Likewise, the following entry creates a 128Kb disk cache in expanded memory:

```
DEVICE=C:\DOS\SMARTDRV.SYS 128   /A
```

Because expanded memory must swap pages to access the cache, an extended memory disk cache will have better performance.

NOTE *Before you can install disk-caching software into extended or expanded memory, the corresponding memory-management software must be active. Place the DEVICE entry for SMARTDRV.SYS in the CONFIG.SYS file after the entry for the memory-management device driver.*

Invoking the SMARTDRV Command

Beginning with DOS version 6, DOS provides the command SMARTDRV.EXE that gives you greater flexibility in controlling the SMARTDRV cache. Most users will place the SMARTDRV command into their AUTOEXEC.BAT file. The format of the SMARTDRV command is as follows:

```
SMARTDRV [[Drive[+|-] ...] [/E:ElementSize] [InitialCacheSize]
    [MinimumCacheSize] [/B:BufferSize] [/C] [/R] [/L] [/Q] [/S]
```

Drive[+|–] specifies the desired disk drive for which you want to enable or disable caching. If you don't specify a plus or minus sign, SMARTDRV performs read caching but not write caching. If you specify a plus sign, SMARTDRV performs both. If you specify a minus sign, SMARTDRV disables caching.

/E:*ElementSize* specifies the amount of data (in bytes) that SMARTDRV moves at one time. You must specify 1,024, 2,048, 4,096 or 8,192 bytes. The default size is 8,192.

InitialCacheSize specifies, in Kb, the size of the cache when SMARTDRV starts.

MinimumCacheSize specifies, in Kb, the size to which applications (such as Windows) wanting extended memory can reduce the cache. When the application ends, the previous cache size will resume.

/B:*BufferSize* specifies, in Kb, the size of SMARTDRV's read-ahead buffer. The default size is 16Kb.

/C Directs SMARTDRV to write all cached (output) data from memory to disk.

/R clears the current cache and restarts SMARTDRV.

/L directs SMARTDRV to load itself into conventional memory, as opposed to upper memory.

/Q directs SMARTDRV not to display error messages as it starts.

/S directs SMARTDRV to display its status information.

Before you can invoke SMARTDRV, you must first install the HIMEM.SYS device driver, which gives DOS access to extended memory. The following SMARTDRV command allocates a 2Mb cache, enabling read and write caching for drive C:

```
C:\> SMARTDRV C+ 2048 <ENTER>
```

Before you turn off your computer, you may want to first flush SMARTDRV's cache to disk using the /C switch, as shown here:

```
C:\> SMARTDRV /C <ENTER>
```

Finally, the following command directs SMARTDRV to display its status information. In general, the status information lets you know just how useful the cache is. A *cache hit* occurs when SMARTDRV finds the data it desires in memory, which eliminates a disk read operation. A *cache miss* occurs when SMARTDRV must read the data from disk.

```
C:\> SMARTDRV /S <ENTER>
Microsoft SMARTDrive Disk Cache version 4.0
Copyright 1991,1992 Microsoft Corp.

Room for 256 elements of 8,192 bytes each
There have been 166 cache hits
   and 379 cache misses

Cache size: 2,097,152 bytes
Cache size while running Windows: 2,097,152 bytes

            Disk Caching Status
drive   read cache   write cache   buffering
--------------------------------------------
  A:       yes           no           no
  B:       yes           no           no
  C:       yes           yes          no

For help, type "Smartdrv /?".
```

Reducing the Number of DOS Disk Buffers

As you learned in Chapter 24, the CONFIG.SYS BUFFERS entry lets you specify the number of DOS disk buffers. If you install a disk cache using SMARTDRV.SYS or SMARTDRV.EXE, you need to reduce the number of DOS disk buffers within CONFIG.SYS. For example, the following entry reduces the number of buffers to 3:

```
BUFFERS=3
```

System Considerations for Performance

Chapter 32 examines DOS memory management in detail. In general, most larger applications will run faster given more memory because the programs can create larger buffers and keep more information in memory, rather than the slower disk.

If you are using DOS 5 or later, DOS gives you many ways to effectively manage your computer's memory to make more memory available to your applications. In addition to looking at DOS 5 or later memory management, Chapter 32 examines extended and expanded memory and explains why extended memory provides better performance.

In Chapter 41 you will learn the MODE command can improve keyboard responsiveness. To begin, hold down a key on your keyboard and note the amount of time DOS waits before it repeats the character for the key depressed. Next, issue the following MODE command:

```
C:\> MODE  CON:  RATE=32  DELAY=1 <ENTER>
```

Hold down the same key once again and note the difference in your keyboard's responsiveness.

As discussed briefly in Chapter 24 with respect to the CONFIG.SYS BREAK entry and again in Chapter 41 with respect to the BREAK command, extended CTRL-BREAK checking is the process of increasing the number of times DOS checks for a user-entered CTRL-BREAK to end a command. By default, DOS checks for a CTRL-BREAK each time it writes to your screen, printer, or auxiliary device (COM1) or when it reads from your keyboard or auxiliary device. Because some programs don't perform these input and output operations often, it may take DOS considerable time to acknowledge a CTRL-BREAK. When you enable extended CTRL-BREAK checking, DOS will check for a CTRL-BREAK every time it performs a function such as opening a file, reading or writing to disk, or even returning the current system date or time to a program. Because DOS tests for CTRL-BREAK more often, DOS will recognize and respond to a CTRL-BREAK much sooner. Unfortunately, because DOS now spends more time testing for a CTRL-BREAK, DOS spends less time letting your programs run. As a result, your system performance decreases.

Unless you are a programmer who is developing and testing programs, you should leave extended CTRL-BREAK checking disabled. Using the following BREAK command, determine the state of extended CTRL-BREAK checking:

```
C:\> BREAK <ENTER>
BREAK is off
```

If your system has extended CTRL-BREAK checking ON, check your CONFIG.SYS for a BREAK=ON entry, or AUTOEXEC.BAT for a BREAK ON command. If you find either, remove the entry from the file and issue the BREAK OFF command.

SUMMARY

Improving System Performance

It doesn't take long for most users to want their computers to work faster. There are several things you can do to improve your system's performance without buying a new computer.

If you open certain files more than once (in DOS 3.3 or later), you can use the FASTOPEN command to store the disk addresses of any files opened since startup. This saves the computer from having to locate each file repeatedly. If you limit the number of files in a directory to those that will fit in one cluster, the directory entry will not become fragmented, and the frequent directory reads will be much faster. If you use a PATH command, putting the most frequently used directories first will eliminate a lot of directory searching. Remove from the PATH directories you seldom use, especially large ones.

If there are files you don't need, such as international setup files or BASIC programs, remove them from your hard disk or set them aside in a subdirectory. If you have enough extended memory for a RAM drive, assign your TEMP files to it—it's much faster than a hard drive.

Fix your fragmented files. If you use DOS 4 or later and don't use SMARTDRV, use BUFFERS for disk I/O. If you can use SMARTDRV (EXE or SYS), it's faster, and you can reduce the number of buffers. In general, use RAM instead of disks when you can—it's much faster.

If your keyboard rate slows you down, use the MODE command to adjust the delay time and repeat speed.

Hands On

If you are using DOS 3.3, place a FASTOPEN command in your AUTOEXEC.BAT file. Most users will find tracking 50 files is sufficient.

```
FASTOPEN C:=50
```

If you are using DOS 4, use the CONFIG.SYS INSTALL entry to load FASTOPEN, as discussed in Chapter 24.

```
INSTALL=C:\DOS\FASTOPEN.EXE  C:=50
```

If you are using DOS 5 or later, place the following LOADHIGH command in your AUTOEXEC.BAT:

```
LOADHIGH C:\DOS\FASTOPEN.EXE  C:=50
```

Next, issue the following PATH command to display the current command path:

```
C:\> PATH <ENTER>
```

For each directory specified in the command path, ask yourself if the directory really contains commands you use on a regular basis. If the answer is no, edit your AUTOEXEC.BAT file and update the PATH command. If you use APPEND to define a data-file search path, ask yourself the same question.

Next, again examine the directories in the current command path. For each directory, ask yourself if there are files you can move from the directory into a lower directory to reduce the number of files DOS must compare while searching through the command path.

Last, use CHDIR to select your commonly used directories and then issue the following CHKDSK command to examine the files for fragmentation.

```
C:\DIRNAME> CHKDSK  *.* <ENTER>
```

If a file is fragmented, CHKDSK will display the following message:

```
FILENAME.EXT Contains n non-contiguous blocks
```

If CHKDSK only displays a few fragmented files, don't worry. However, if a large percentage of your files are fragmented, you may want to turn to Chapter 41 and learn how to correct the fragmentation.

Review

1. Given a disk with 4096-byte clusters, what is the maximum number of files you would ideally like a directory to contain? Why?
2. FASTOPEN improves system performance by tracking the disk locations of your commonly used files. If a setting of 50 FASTOPEN table entries is good, why isn't 999 better?
3. Why does a large number of entries in the command path decrease system performance?

4. Which is preferable, DOS BUFFERS or a SMARTDRV cache? Why?
5. How do fragmented files decrease system performance?

Answers

1. To determine the number of entries you should place in a directory, divide the disk's cluster size by 32 (the number of bytes in a directory entry). For a 4096-byte cluster, the ideal number of directory entries is 128. If you store more than 128 directory entries, the directory will require two or more clusters, very likely becoming fragmented, which will decrease your system performance on every directory operation.
2. FASTOPEN's table of directory entries is not without cost. Each entry FASTOPEN tracks requires 48 bytes of memory. If you created a FASTOPEN table for 999 entries, you would quickly consume almost 48Kb of memory, perhaps better used by a different program. In addition, the larger FASTOPEN's table becomes, the longer it will take FASTOPEN to search the table, decreasing your system performance. The goal of FASTOPEN is to track your most commonly and recently used files. Very few users reuse 999 files on a regular basis.
3. If the number of entries in your command path is large, DOS must spend considerable time searching each directory for external commands and batch files. You can reduce this overhead some by placing the directories most likely to contain commonly used commands at the start of the command path.
4. If you have extended or expanded memory, the SMARTDRV.SYS cache will provide better performance. First, it uses intelligent software to reduce the cache buffer searching time. Second, because it resides in your computer's extended or expanded memory, it does not consume conventional memory better used by an application.
5. A fragmented file is a file whose contents are in clusters spread out across your disk. Fragmented files decrease system performance because they increase rotational disk delays and increase the number of disk head seek operations required to access the file.

Advanced Concepts

You read in Chapter 28 that each time you delete a file, DOS does not actually erase the file's contents from disk; rather, DOS marks the file's directory entry and disk clusters as available for reuse by another file. Assume, for example, a directory on your disk contains the files ONE, TWO, and THREE. The directory contents might contain the entries shown here:

ONE	1024	7-04-93	12:01P
TWO	2048	7-04-93	12:02P
THREE	3072	7-04-93	12:03P

If you later delete file TWO, DOS will mark the file's directory listing entry as available for reuse by setting the first character of the filename to a lowercase sigma (σ), as shown here:

ONE	1024	7-04-93	12:01P
σWO	2048	7-04-93	12:02P
THREE	3072	7-04-93	12:03P

If you later create the file FOUR in the directory, DOS will reuse the first available directory entry, as shown here:

ONE	1024	7-04-93	12:01P
FOUR	2048	7-05-93	12:04P
THREE	3072	7-04-93	12:03P

Next, assume you have a directory containing 30 files with the LTR extension and you delete the files. As before, DOS marks each directory entry as available for reuse as shown in Figure 31-1. If you delete a large number of files, your directory will contain many entries marked available for reuse.

The problem with having these unused entries in the directory is every time DOS searches the directory, DOS has to search through these unused entries. If you never create additional files in the directory, these entries will remain unused, and a continual source of directory search overhead.

If you are using DOS 5 or later, you can use the following UNDELETE command to determine the number of unused directory entries.

```
C:\> UNDELETE  *.*  /DOS  /LIST <ENTER>
```

FILENAME.ONE	1024	7-04-93	12:01P
FILENAME.TWO	2048	7-04-93	12:02P
σNE.LTR	1024	7-04-93	12:01P
σWO.LTR	2048	7-04-93	12:02P
σHREE.LTR	3092	7-04-93	12:03P
.	.	.	.
.	.	.	.
.	.	.	.
σHIRTY.LTR	30720	7-04-93	12:30P
FILENAME.THR	3092	7-04-93	12:03P

Figure 31-1. *Unused directory entries available for reuse*

If you want to get rid of the unused entries, use the following BACKUP command to back up the files in the directory to floppy.

```
C:\DIRNAME>  BACKUP  *.*  A:  /S <ENTER>
```

When the backup operation completes, delete the files and subdirectories from the directory. Next, use RMDIR to remove the directory itself. Next, use MKDIR to create the directory once again and CHDIR to select the directory. Using the following RESTORE command you can restore the directory and its files, minus the unused entries:

```
C:\DIRNAME>  RESTORE  A:  *.*  /S <ENTER>
```

Key Terms

Disk cache A large, fast, specialized disk buffer that improves disk I/O performance even more effectively than CONFIG.SYS BUFFERS.

Performance The efficiency of your computer's hardware and software.

Chapter 32

Understanding DOS Memory Management

As computer programs become more powerful and complex, the list of instructions needed to implement the programs has become very large. In fact, many programs are quickly outgrowing the 640Kb memory limits of the IBM PC. This chapter examines the different solutions that help you squeeze the most information into your computer's memory. In particular, you will learn the differences between, and how to use, conventional, reserved, and high memory. In addition, this chapter fully explains extended and expanded memory. If this book has a most difficult chapter, this may be it. However, if this book has a most valuable chapter, this one is definitely in the running. The figures in this chapter are worth more than a thousand words, so pay close attention to them.

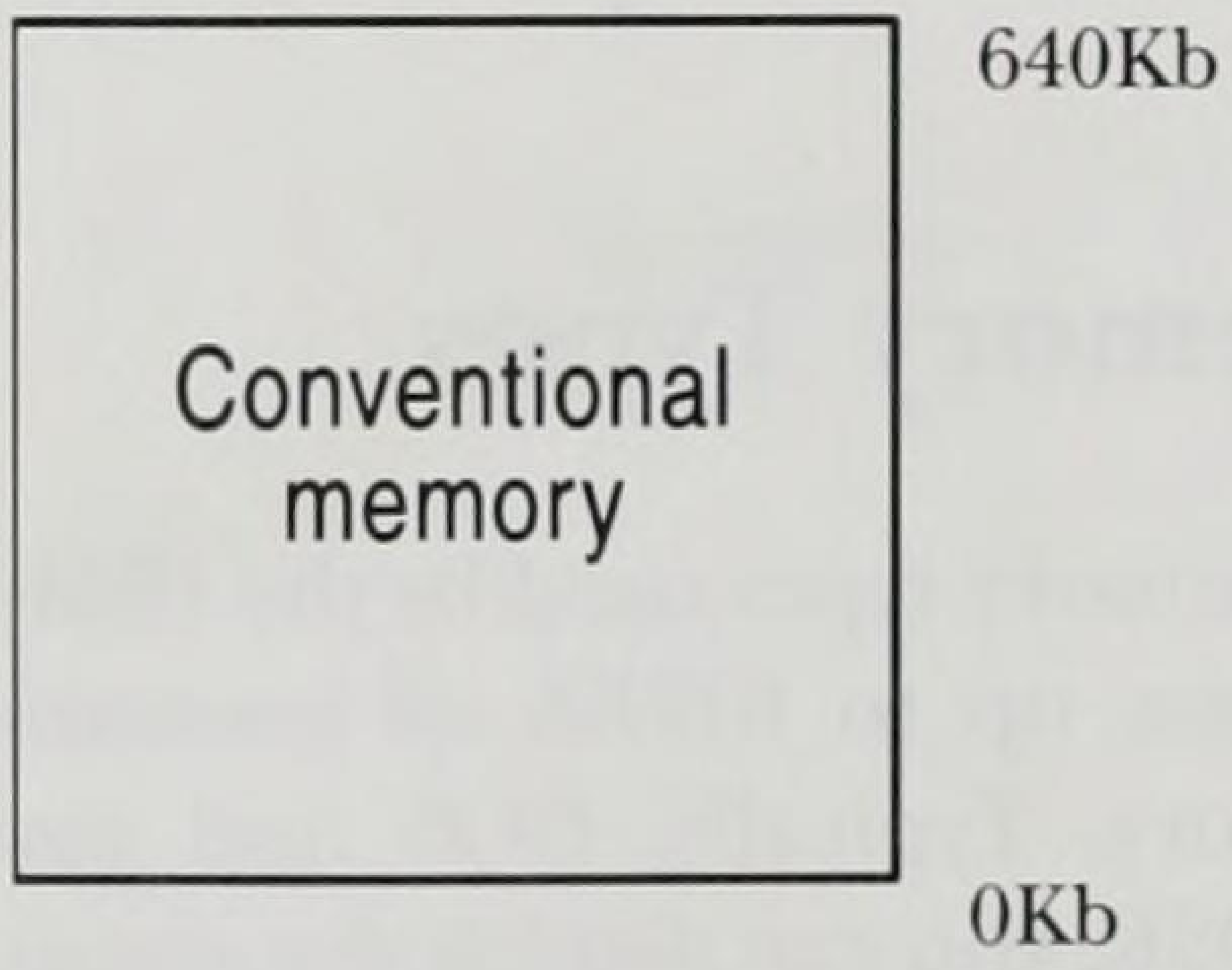
Understanding Memory Types

There are three primary memory types used by the IBM PC and PC-compatibles. To begin, most computers have up to 640Kb of *conventional memory*, which is your computer's primary memory. Typically, DOS and your programs reside in this memory area. Using CHKDSK, you can display the amount of conventional memory in your computer.

```
C:\> CHKDSK <ENTER>

Volume DOS          created 02-21-1993 8:46a
Volume Serial Number is 168C-5277

 200065024 bytes total disk space
     81920 bytes in 3 hidden files
    462848 bytes in 108 directories
  44785664 bytes in 2036 user files
 154734592 bytes available on disk

      4096 bytes in each allocation unit
     48844 total allocation units on disk
     37777 available allocation units on disk

    655360 total bytes memory
    562688 bytes free
```

When the IBM PC was first released in 1981, computers typically had 64Kb to 256Kb of conventional memory. As memory became more affordable, the amounts grew to 512Kb and then to 640Kb. Depending on your computer's age, the amount of conventional memory will fall somewhere in the range 64Kb to 640Kb.

When your computer starts, DOS loads itself into the lower part of conventional memory. If you install memory-resident programs, DOS places the programs in the next available lower memory areas. The amount of memory that remains determines the largest program your computer can run. This illustration shows the conventional memory layout:



Although conventional memory cannot exceed 640Kb, the original IBM PC can address up to 1 megabyte of memory. If you subtract 640Kb from 1Mb, you'll find the PC has a 384Kb gap between the end of conventional memory and the highest memory location the IBM PC can access. As shown in Figure 32-1, your computer uses a portion of this memory to interface to your video display and a portion for the ROM-BIOS routines, leaving a portion of the 384Kb region unused.

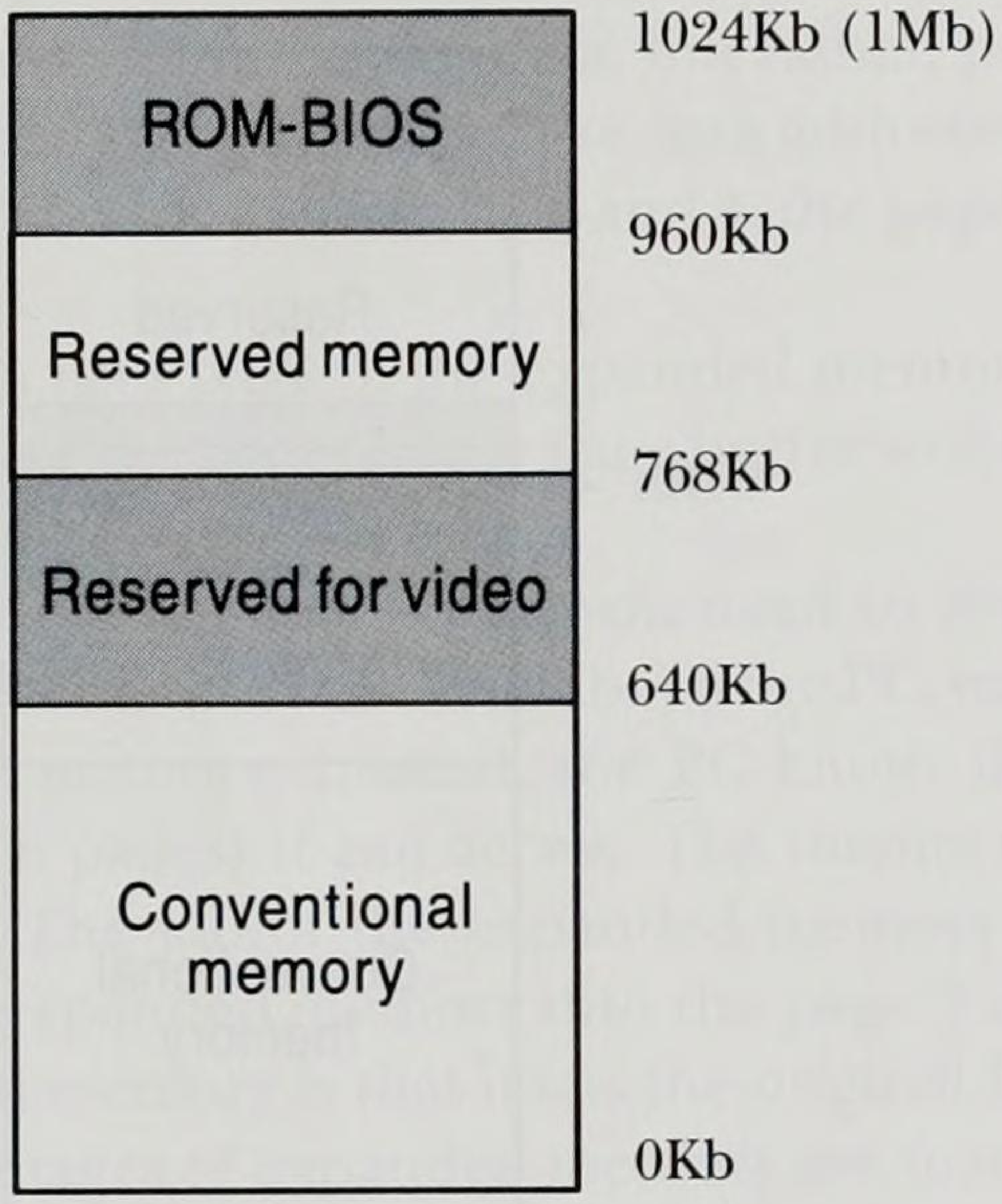


Figure 32-1. *How your computer uses the 384Kb region above conventional memory*

Expanded Memory

Although a portion of the 384Kb region above conventional memory was unused, the region wasn't large enough to be of much use on its own. However, it didn't take hardware developers very long to figure out a way to use this small memory region to access memory outside of the PC's 1Mb limitation. *Expanded memory* combines hardware memory boards and powerful software called an *EMS driver* to expand the amount of memory DOS can use from 640Kb to up to 32Mb!

Here's how the expanded memory specifications (EMS) software works. There are two major versions of EMS software, LIM (Lotus Intel Microsoft) EMS version 3.2 and version 4.0. For simplicity, the initial discussion is based on version 3.2. Later in this chapter, you will learn how version 4.0 differs.

First, the EMS software uses a 64Kb region of memory in the reserved memory above 640Kb and below 1Mb. As shown in Figure 32-2, the expanded memory software divides this region into four 16Kb regions, called *page buffers.*

Next, when you run a program that supports expanded memory, the program is loaded into conventional memory, and its data is loaded into the memory on the expanded memory board. Such data might include a very large spreadsheet. The program must divide its data into 16Kb sections called *pages,* as shown in Figure 32-3.

When the PC needs to access data contained in an expanded memory page, the expanded memory software maps the address of a 16Kb page containing data to an address of one of the page buffers that the PC can access. For example, when the program first starts, the expanded memory driver maps the page of data from

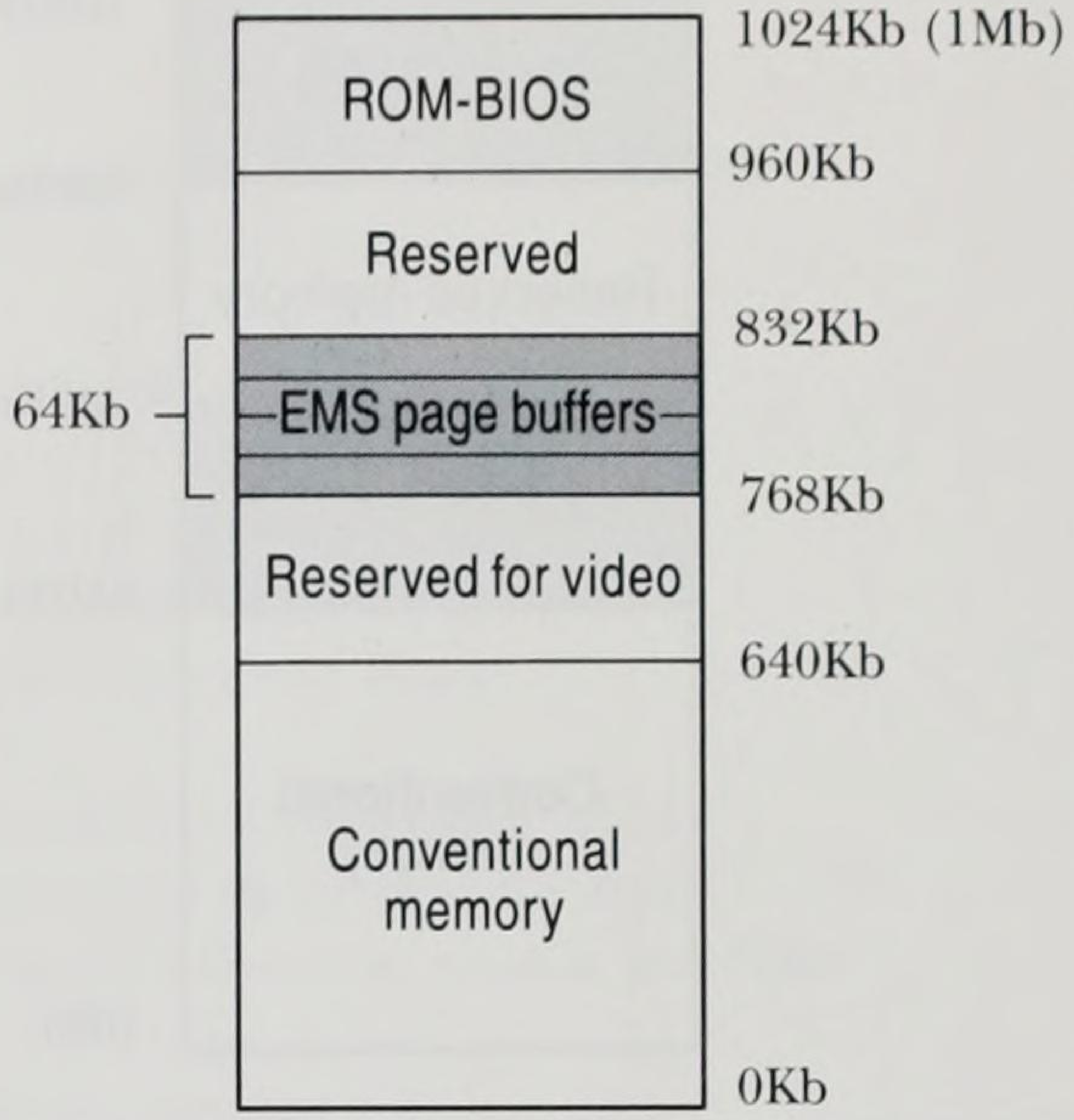

1024Kb (1Mb)
ROM-BIOS
960Kb
Reserved
832Kb
64Kb
EMS page buffers
768Kb
Reserved for video
640Kb
Conventional memory
0Kb

Figure 32-2. *The expanded memory software uses a 64Kb region in the reserved memory region above 640Kb*

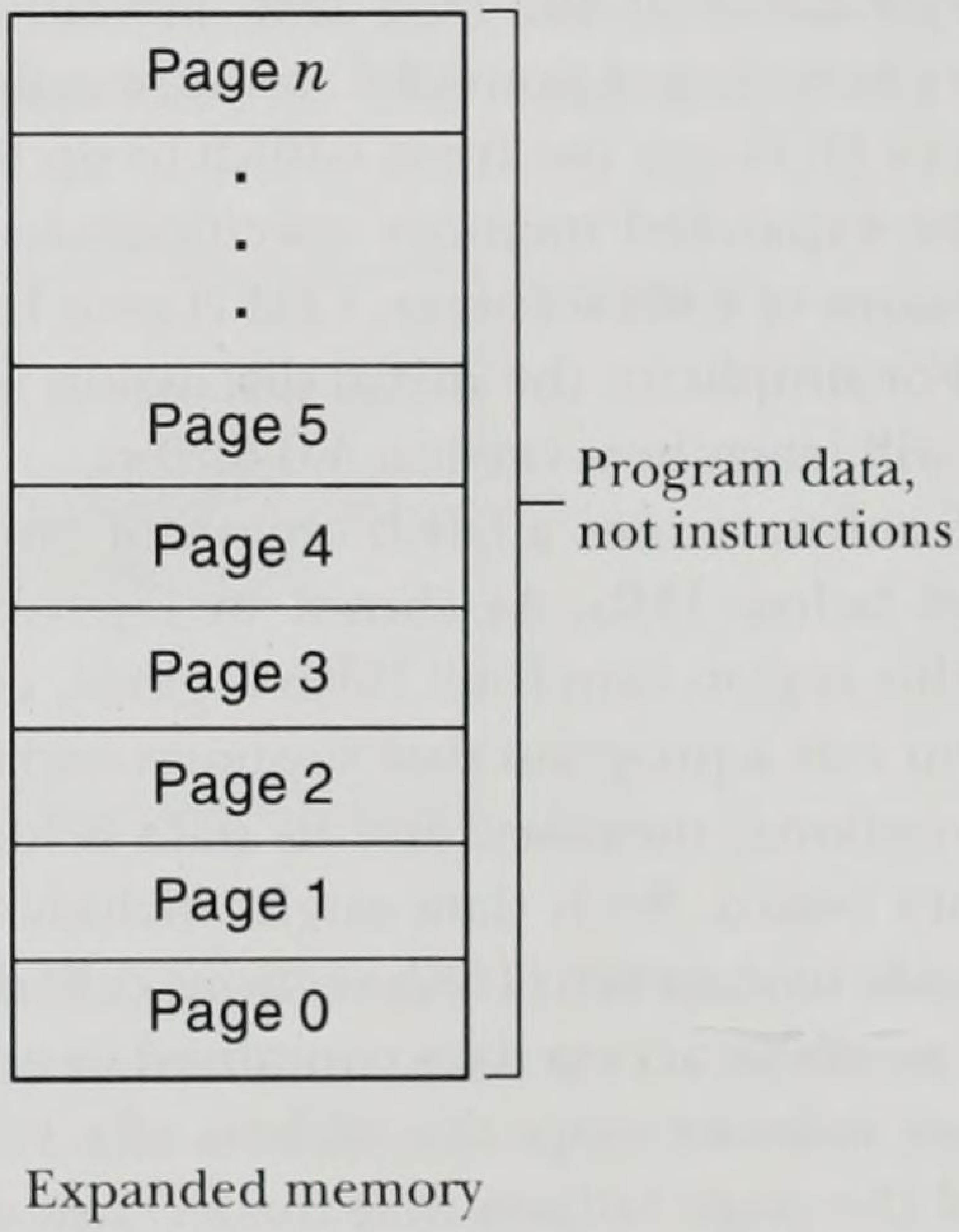

Page n
.
.
.
Page 5
Page 4
Page 3
Page 2
Page 1
Page 0
Program data, not instructions
Expanded memory

Figure 32-3. *A program divides its expanded memory data into 16Kb regions called pages*

expanded memory into a page buffer as shown in Figure 32-4. Note that the expanded memory software does not move the data, but rather, performs memory address manipulation behind the scenes to make the data addressable to the PC.

Assuming the program uses data in pages 1, 2, 3, and 4, the page buffers will map the pages shown in Figure 32-5.

Assuming the program then uses page 5, the expanded memory driver changes the address mapping from one of the pages in the page buffer so it can map page 5's address, as shown in Figure 32-6.

To understand how expanded memory works, you need to remember that the original IBM PC could not access memory above 1Mb. To the PC, expanded memory is not addressed like standard memory. Instead, the PC knows there are 64Kb of memory (containing four 16Kb pages) it can access. The memory itself resides on an expanded memory board. The job of the expanded memory driver is to map information contained in the expanded memory into the page frame.

The advantage of expanded memory is that it lets the original IBM PC break the 640Kb boundary. The disadvantages of expanded memory are that not all programs support it, and those that do may execute slowly because of the continual time-consuming address mappings required to map different pages in and out of the page buffers.

To use expanded memory, you will need an expanded memory board (or a board that simulates expanded memory) and a device driver for the board. You must install the device driver using the CONFIG.SYS DEVICE entry discussed in Chapter 24. The

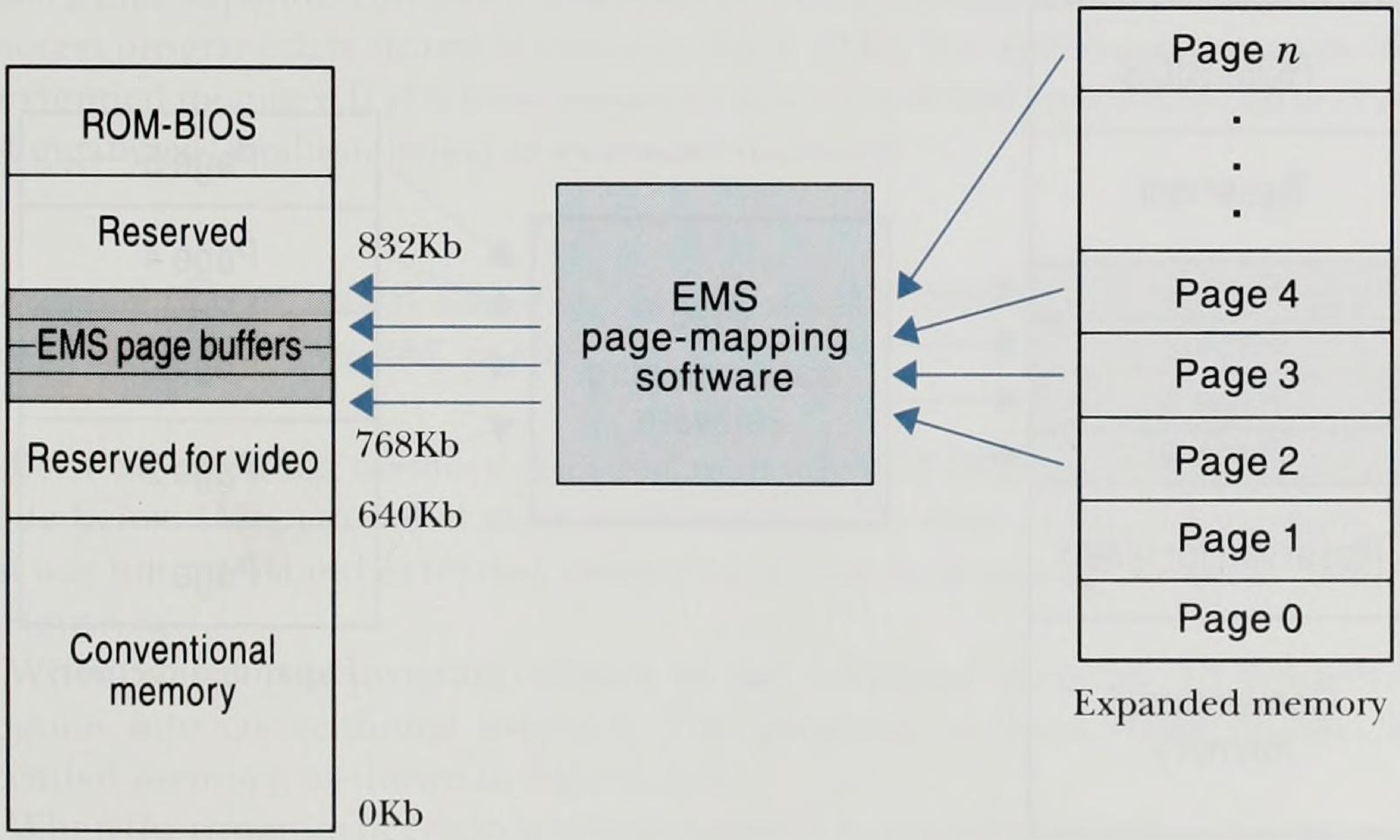


Figure 32-4. *The expanded memory software maps pages of expanded memory into the page buffers the PC can access*

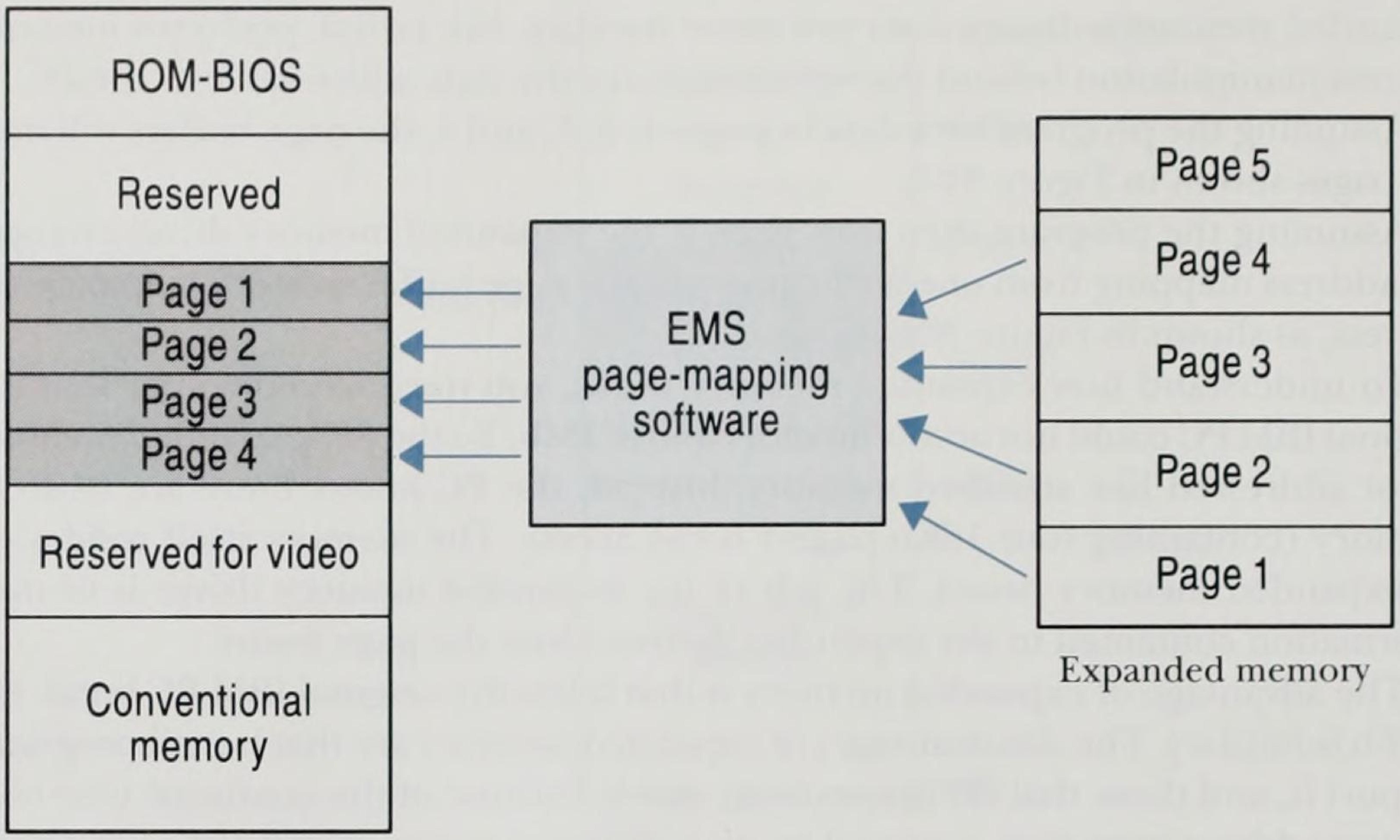

ROM-BIOS
Reserved
Page 1
Page 2
Page 3
Page 4
Reserved for video
Conventional memory
EMS page-mapping software
Page 5
Page 4
Page 3
Page 2
Page 1
Expanded memory

Figure 32-5. *The page buffers can hold up to four pages at one time*

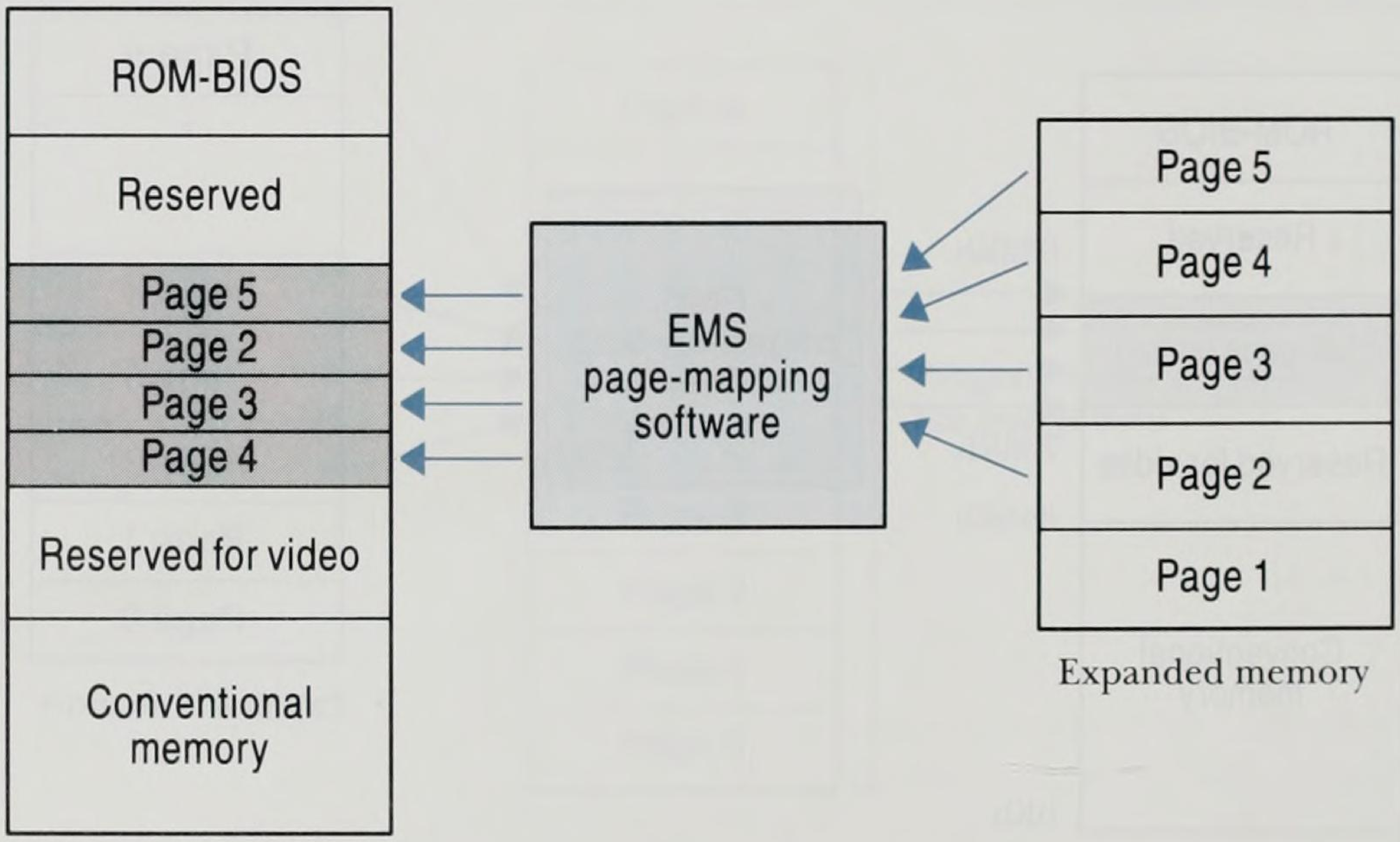

ROM-BIOS
Reserved
Page 5
Page 2
Page 3
Page 4
Reserved for video
Conventional memory
EMS page-mapping software
Page 5
Page 4
Page 3
Page 2
Page 1
Expanded memory

Figure 32-6. *As a program executes, the expanded memory software changes address mapping to access different pages*

device driver should be on a floppy disk that accompanies your memory board from the manufacturer.

The companies that initially drove the expanded memory development are Lotus, Intel, and Microsoft (LIM). The expanded memory boards and software you buy should meet the LIM expanded memory specifications (LIM EMS).

How LIM EMS Version 4.0 Differs LIM EMS version 3.2 supported four fixed-size 16Kb page frames. The maximum amount of expanded memory a program could access using version 3.2 was 8Mb. The page frames had to map to a memory location above 640Kb.

LIM EMS 4.0 removed many restrictions of page size and page location found in version 3.2. To maintain compatibility with version 3.2, EMS 4.0 requires four fixed-size 16Kb page frames. After that, version 4.0 lets you create any number of page frames of any size, letting the pages map to any memory location. In addition, version 4.0 increases the amount of expanded memory a program can access to 32Mb. EMS version 4.0 also includes many capabilities required in a multitasking environment.

Extended Memory

Although expanded memory broke the IBM PC's 640Kb memory restriction, the overhead of mapping pages in and out of the page buffers left hardware developers looking for a more efficient way of accessing memory above 1Mb. When IBM introduced the PC AT (80286) in 1984, they also introduced the notion of *extended memory*. Like expanded memory, extended memory combines hardware and software to access program data stored in memory above 1Mb. The 80286 can use up to 16Mb of extended memory. If you have money to burn, the 80386 and 80486 can access up to 4 gigabytes (4 billion bytes) of extended memory!

The original IBM PC (8088) cannot use extended memory. To use extended memory, you must be using an 80286, 386, or 486.

NOTE

Because extended memory does not swap pages in and out of page buffers that reside below 1Mb, extended memory is much faster than expanded memory. The best way to understand extended memory is as one large storage location, as shown in Figure 32-7.

When you run a program written to use extended memory, DOS loads the program into conventional memory. The program, in turn, loads its data into extended memory, as shown in Figure 32-8.

When the program needs to access data stored in extended memory, the extended memory software lets the program access the data directly, without the page mapping done by expanded memory.

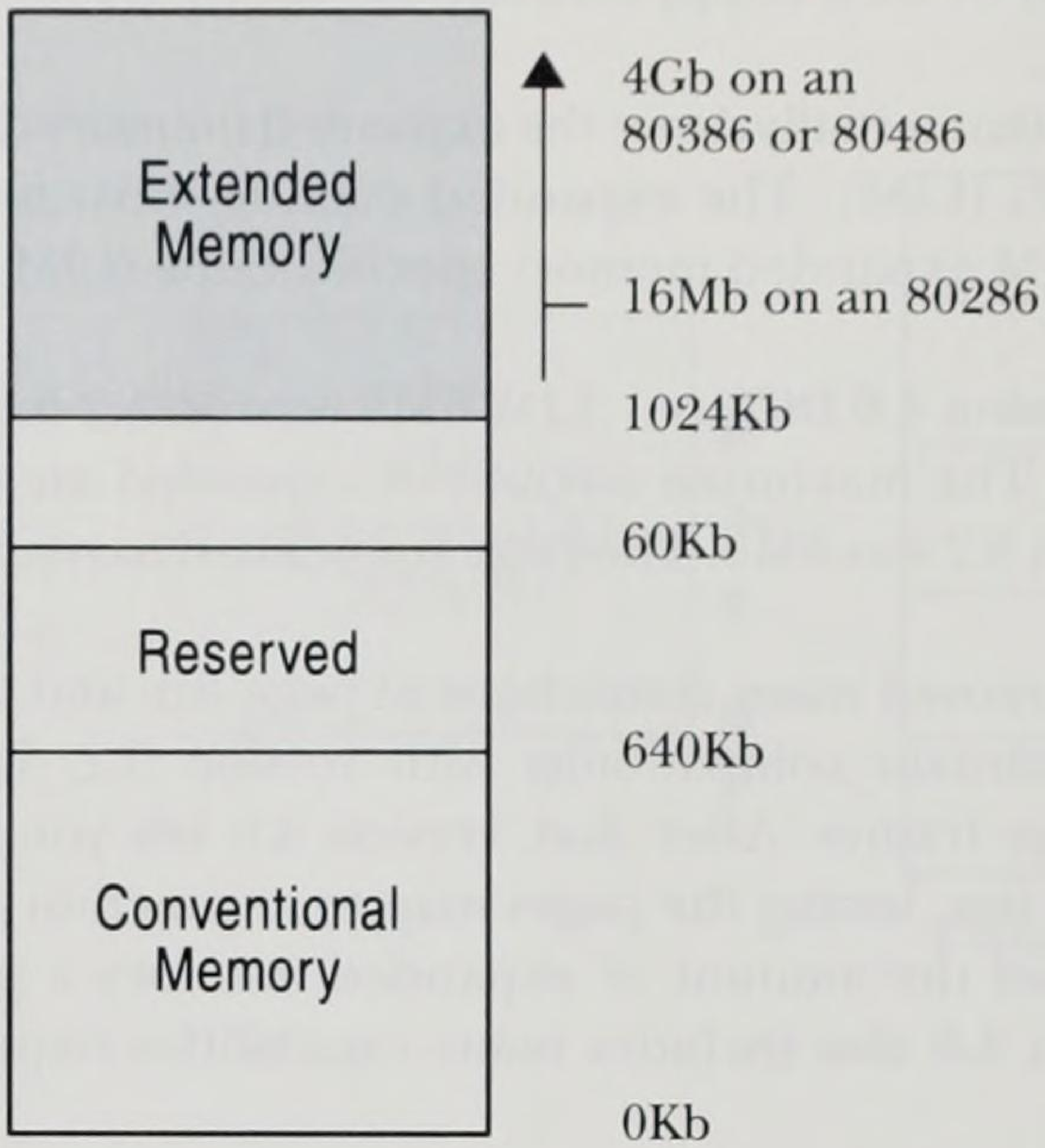

Extended Memory
4Gb on an 80386 or 80486
16Mb on an 80286
1024Kb
60Kb
Reserved
640Kb
Conventional Memory
0Kb

Figure 32-7. *Extended memory is one large memory region that resides above 1Mb*

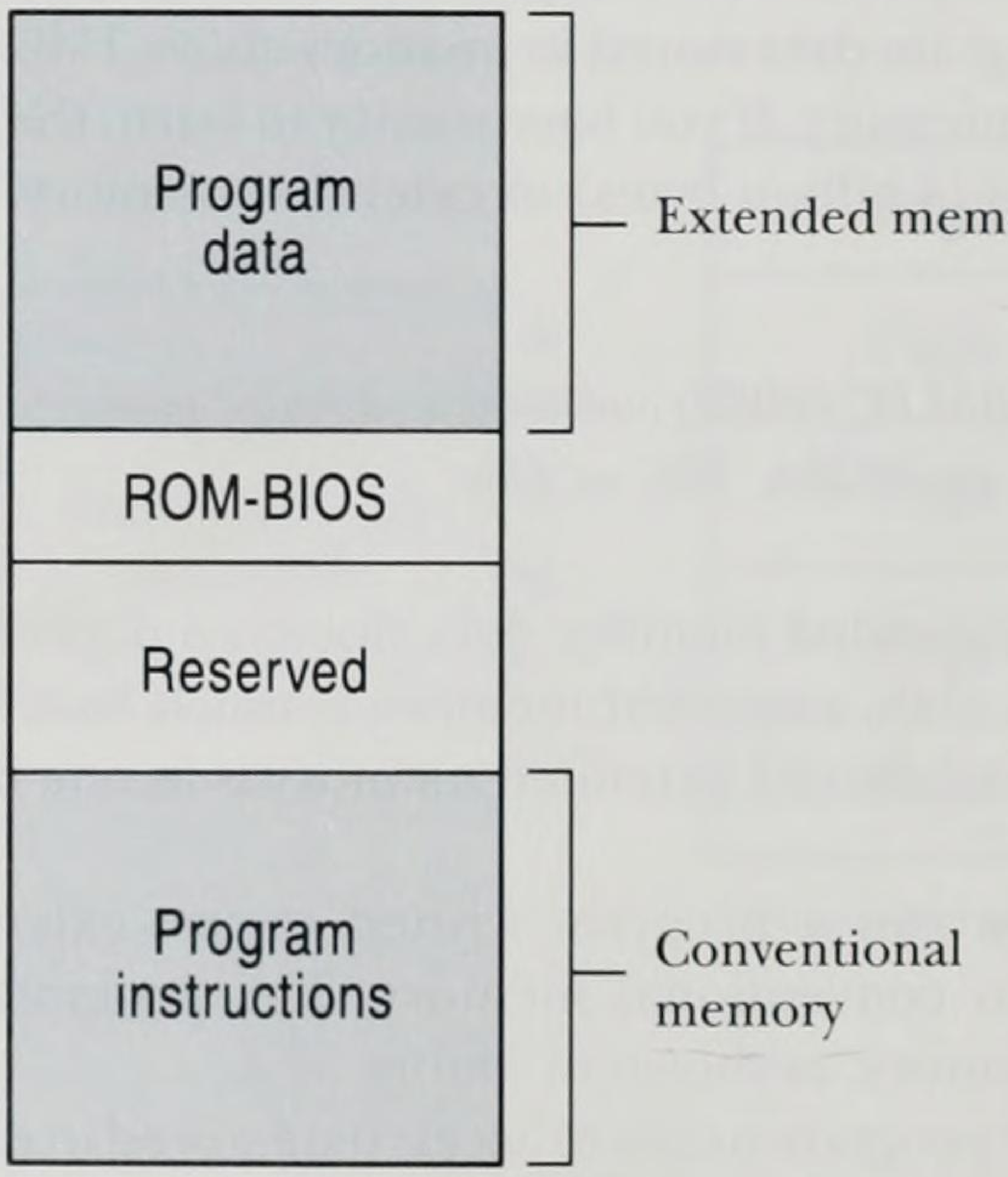

Program data
Extended memory
ROM-BIOS
Reserved
Program instructions
Conventional memory

Figure 32-8. *DOS loads programs that support extended memory into conventional memory; the program, in turn, loads its data into extended memory*

Extended memory requires a software device driver that lets DOS use the memory. If you are using DOS 5 or later or Microsoft Windows 3.0, the device driver HIMEM.SYS provides DOS access to extended memory. Using the CONFIG.SYS DEVICE entry, you can install the driver as follows:

```
DEVICE=C:\DOS\HIMEM.SYS
```

If you are not using DOS 5 or later, change the directory path to WINDOWS, as shown here:

```
DEVICE=C:\WINDOWS\HIMEM.SYS
```

How We Use Extended and Expanded Memory

Although extended and expanded memory let DOS break the 640Kb boundary, both techniques have a big limitation. In most cases, you can only store a program's data above 1Mb. The program's instructions must still reside below 1Mb. The 80286, 386, and 486 have two modes of operation. In DOS, these computers run in *real mode,* which means program instructions must reside below 1Mb. In other operating systems such as OS/2, UNIX, or even Microsoft Windows (running in 386 Enhanced mode) your computer runs in *protected mode,* which lets a program's data and instructions reside in extended memory.

Unfortunately, because DOS runs in real mode, program instructions must reside below 1Mb. Even if your computer has a large amount of extended memory, many applications still don't take advantage of it. Therefore, users most commonly use extended and expanded memory for large disk buffers called disk caches or for fast electronic RAM drives, as discussed next.

Creating a RAM Drive

If you have unused memory available, conventional, expanded, or extended, DOS lets you use the memory to create an electronic *RAM drive.* Because RAM drives reside in your computer's fast electronic memory, they are many times faster than a mechanical hard disk. When you create a RAM drive, DOS assigns to the drive the first unused drive letter, such as D or E. Because of their tremendous speed, many users use RAM drives to store temporary files. However, because the files are stored in your computer's electronic memory, the information RAM drives contain is lost when you turn off your computer's power or restart DOS. RAM drives, therefore, are best suited for temporary files.

The RAMDRIVE.SYS device driver lets you install a RAM drive. To install the device driver, you must use the CONFIG.SYS DEVICE entry. The format of RAMDRIVE.SYS is shown in the following example.

```
DEVICE=C:\DOS\RAMDRIVE.SYS [[DiskSize SectorSize]
  [DirectoryEntries]] [/A|/E]
```

DiskSize specifies the size of the disk in kilobytes. If you don't specify a value, the default size is 64Kb. RAMDRIVE.SYS lets you specify a size from 16 through 4096Kb (4 megabytes). *SectorSize* specifies the RAM drive's sector size. The default sector size is 512 bytes. RAMDRIVE.SYS lets you specify sector sizes of 128, 256, or 512 bytes. *DirectoryEntries* specifies the number of files and directories the RAM drive can store in its root directory. Like all disk drives, RAM drives restrict the number of files you can store in the root directory. By default, the RAM drive's root directory can hold 64 entries. RAMDRIVE.SYS lets you specify from 2 to 1024 entries.

By default, RAMDRIVE.SYS creates the RAM drive in conventional memory. The /E switch directs RAMDRIVE.SYS to create the RAM drive in extended memory. Likewise, the /A switch directs RAMDRIVE.SYS to use expanded memory.

The following CONFIG.SYS entry creates a 64Kb RAM drive in conventional memory:

```
DEVICE=C:\DOS\RAMDRIVE.SYS
```

When you start your computer, your screen will display a message similar to the following, possibly using a different disk drive letter or disk size.

```
Microsoft RAMDrive version n.nn virtual disk D:
    Disk size: 64k
    Sector size: 512 bytes
    Allocation unit: 1 sectors
    Directory entries: 64
```

To access the drive, use the drive letter D, as shown here:

```
C:\> DIR  D: <ENTER>
```

Likewise, the following entry creates a 512Kb RAM drive in extended memory:

```
DEVICE=C:\DOS\RAMDRIVE.SYS  512   /E
```

DOS 5 or later uses the TEMP environment entry, discussed in Chapter 19, to determine where it should create its temporary files during I/O redirection operations. By assigning TEMP to your fast RAM drive, you can improve your system performance.

The information stored in a RAM drive is lost when you turn off your computer or restart DOS. If you need the files the RAM drive contains, make sure you copy the files back to your hard disk or floppy before you turn your computer off.

REMEMBER

Before you can install a RAM drive into extended or expanded memory, the corresponding memory-management software must already be active. Place the RAMDRIVE.SYS DEVICE entry in CONFIG.SYS after the DEVICE entry for your extended memory or expanded memory device driver.

NOTE

Viewing Your Memory Usage

CHKDSK displays statistics about your conventional memory usage. If you are using DOS 4 or later, the MEM command displays detailed information about conventional, expanded, and extended memory usage. When you invoke MEM in DOS 4, your screen will display output similar to the following:

```
C:\> MEM <ENTER>

    655360 bytes total memory
    655360 bytes available
    521440 largest executable program size

   1441792 bytes total extended memory
   1441792 bytes available extended memory
```

In DOS 5, MEM displays

```
C:\> MEM <ENTER>

    655360 bytes total conventional memory
    655360 bytes available to MS-DOS
    606480 largest executable program size

    917504 bytes total EMS memory
    524288 bytes free EMS memory
```

```
1441792 bytes total contiguous extended memory
      0 bytes available contiguous extended memory
 928768 bytes available XMS memory
        MS-DOS resident in High Memory Area
```

The first three lines provide information about your conventional memory usage. The lines that follow summarize your expanded and extended memory usage.

Finally, in DOS 6, the MEM command displays the following:

```
C:\> MEM <ENTER>

Memory Type          Total =  Used  +  Free
----------------    ------   ------   ------
Conventional          640K      73K     567K
Upper                   0K       0K       0K
Adapter RAM/ROM       384K     384K       0K
Extended (XMS)       3072K    2552K     520K
----------------    ------   ------   ------
Total memory         4096K    3009K    1087K

Total under 1 MB      640K      73K     567K

EMS is active.
Largest executable program size       567K  (580896 bytes)
Largest free upper memory block         0K       (0 bytes)
MS-DOS is resident in the high memory area.
```

To help you better understand how DOS is using your memory, MEM provides the /DEBUG switch. Software device drivers and memory-resident programs remain present in your computer's memory until you turn off your computer or restart DOS. If you include the /DEBUG switch, MEM will display the memory address and size of each program or software driver.

```
C:\> MEM  /DEBUG <ENTER>

Conventional Memory Detail:

  Segment                 Size          Name          Type
  -------          ----------------  -----------  --------
   00000              1039    (1K)                Interrupt Vector
   00040                271    (0K)               ROM Communication Area
   00050                527    (0K)               DOS Communication Area
   00070               2656    (2K)  IO           System Data
                                        CON       System Device Driver
                                        AUX       System Device Driver
                                        PRN       System Device Driver
                                        CLOCK$    System Device Driver
                                        A: - C:   System Device Driver
                                        COM1      System Device Driver
                                        LPT1      System Device Driver
                                        LPT2      System Device Driver
                                        LPT3      System Device Driver
                                        COM2      System Device Driver
                                        COM3      System Device Driver
                                        COM4      System Device Driver
   00116               5072    (4K)  MSDOS        System Data
   00253              34688   (33K)  IO           System Data
                        608    (1K)     SETVERXX Installed Device=SETVER
                       1088    (1K)     XMSXXXX0 Installed Device=HIMEM
                       8064    (8K)     EMMXXXX0 Installed Device=EMM386
                       4192    (4K)     CON      Installed Device=ANSI
                      10000   (10K)     MSCD001  Installed Device=SBPCD
                       1184    (1K)     D:       Installed Device=RAMDRIVE
                       2464    (2K)     _doubleB Installed Device=SMARTDRV
                       1488    (1K)               FILES=30
                        720    (1K)               FCBS=12
                        512    (1K)               BUFFERS=4
                       1152    (1K)               LASTDRIVE=M
                       3008    (3K)               STACKS=9,256
   00ACB                 80    (0K)  MSDOS        System Program
   00AD0               2640    (2K)  COMMAND      Program
   00B75                 80    (0K)  MSDOS        – Free –
   00B7A                528    (0K)  COMMAND      Environment
   00B9B                 96    (0K)  MSDOS        – Free –
   00BA1              26816   (26K)  SMARTDRV     Program
   0122D                240    (0K)  MEM          Environment
   0123C              87376   (85K)  MEM          Program
   02791             493296  (481K)  MSDOS        – Free –

Memory Summary:
```

```
Type of Memory          Size         =      Used         +     Free
----------------  -----------------   -----------------   ------------
Conventional        655360   (640K)      74272    (73K)  581088   (567K)

Upper                    0     (0K)          0     (0K)       0     (0K)
Adapter RAM/ROM     393216   (384K)     393216   (384K)       0     (0K)
Extended (XMS)     3145728  (3072K)    2613248  (2552K)  532480   (520K)
----------------  -----------------   ----------------- ----------------
Total memory       4194304  (4096K)    3080736  (3009K) 1113568  (1087K)

Total under 1 MB    655360   (640K)      74272    (73K)  581088   (567K)

EMS is active.
Memory accessible using Int 15h              0     (0K)
Largest executable program size         580896   (567K)
Largest free upper memory block              0     (0K)
MS-DOS is resident in the high memory area.

XMS version  3.00; driver version  3.09
EMS version  4.00
```

If you are using DOS 6, you can invoke MEM with the /FREE list to display your system's available memory.

```
C:\> MEM /FREE <ENTER>

Free Conventional Memory:

  Segment         Size
  -------   ----------------
   00A3C           80    (0K)
   00A62          240    (0K)
   00A71        87376   (85K)
   01FC6       525200  (513K)

  Total Free: 612896  (599K)

Free Upper Memory:

  Region   Largest Free     Total Free      Total Size
  ------  --------------  --------------  --------------
```

```
     1     11264  (11K)     11312  (11K)     11312  (11K)
     2    116528 (114K)    116528 (114K)    143344 (140K)
```

Also, again in DOS 6, you can use MEM's /MODULE switch to determine how much memory a specific program is using. Invoke MEM with the program name (no extension) only. The following command lists the amount of memory currently consumed by COMMAND.COM:

```
C:\> MEM /MODULE COMMAND <ENTER>

COMMAND is using the following memory:

  Segment  Region         Size         Type
  -------  ------  ----------------  --------
   00AD0              2640     (3K)  Program
   00B7A               528     (1K)  Environment
                   ----------------
  Total Size:         3168     (3K)
```

DOS 5 or later Memory Management

Each time your system starts, DOS loads itself into conventional memory. If you are using DOS 5 or later, you can direct DOS to load part of itself into the high memory area that is the first 64Kb of extended memory, as shown in Figure 32-9.

By loading DOS into the high memory area, you free up additional conventional memory for your programs. To load DOS into the high memory area, you must do the CONFIG.SYS DOS entry as follows:

```
DOS=HIGH
```

Before you can load DOS into the high memory area, the extended memory-management software must be active. Place the DOS entry in CONFIG.SYS after the entry that installs the extended memory device driver.

NOTE

The 384Kb memory area above conventional memory and below 1Mb is often called *reserved memory*. As you saw earlier, expanded memory makes use of 64Kb of this area. In DOS 5 or later, you can use portions of reserved memory called *upper memory blocks* (*UMBs*) for software device drivers and memory-resident DOS com-

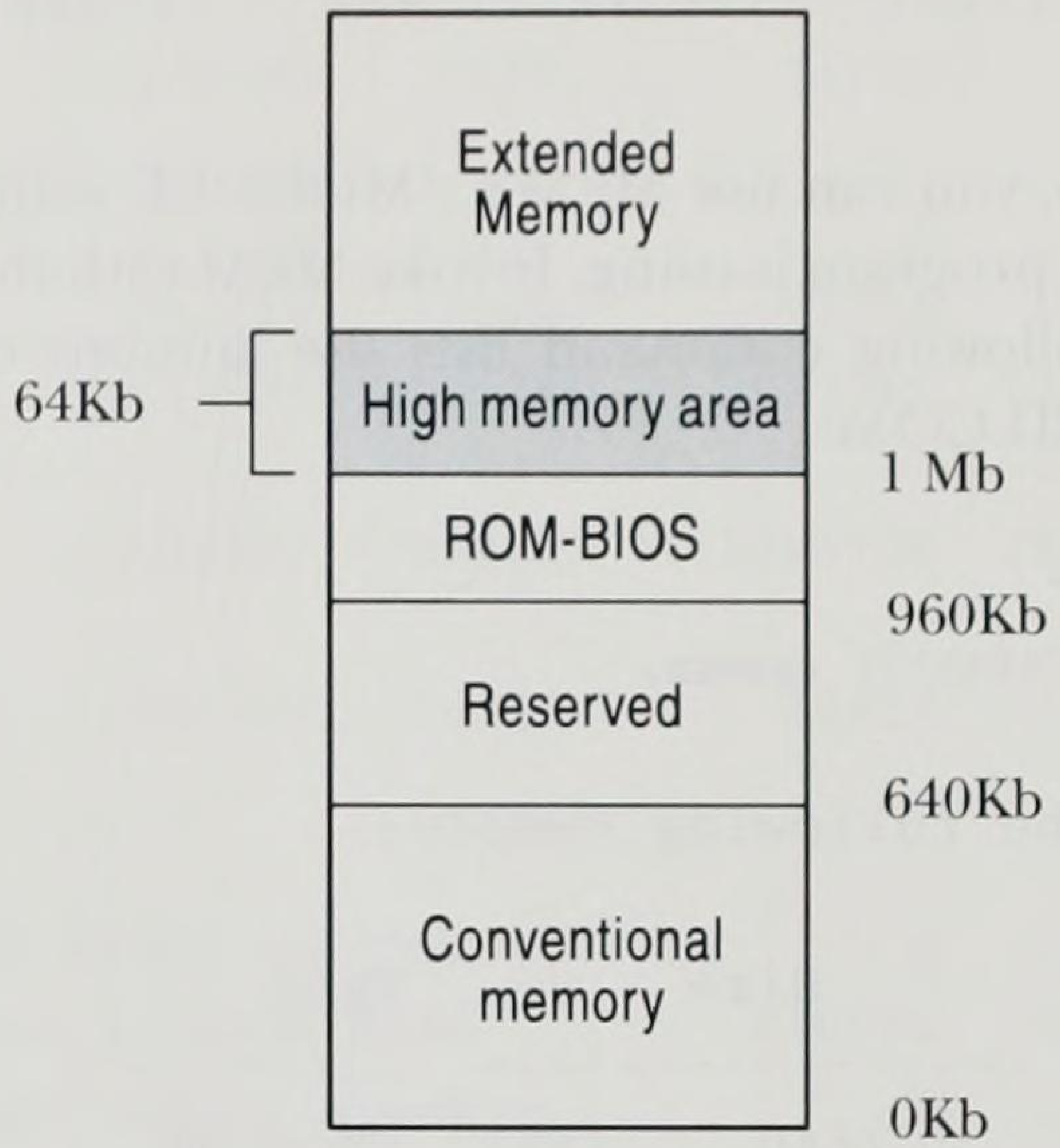


Figure 32-9. *In DOS 5 or later, you can load DOS into the high memory area (the first 64Kb of extended memory)*

mands. To use reserved memory, you must have a 386 or 486 computer and then include UMB in the CONFIG.SYS DOS entry, as shown here:

```
DOS=UMB
```

The UMB entry does not actually use reserved memory. It simply tells DOS you will later use it with the CONFIG.SYS DEVICEHIGH entry or the LOADHIGH command.

An upper memory block is a 64Kb region in reserved memory. By using reserved memory for software drivers or memory-resident commands, you free up conventional memory. The following CONFIG.SYS entry tells DOS you will later use reserved memory, and you want DOS to load itself in high memory.

```
DOS=UMB,HIGH
```

If you aren't using extended memory, you can direct DOS to load itself in conventional memory, while telling DOS you will later use reserved memory, as shown here:

```
DOS=UMB,LOW
```

Last, if you use NOUMB, DOS will not let you later use reserved memory.

In Chapter 24 you learned the CONFIG.SYS DEVICE entry installs a software device driver. DOS places the corresponding software into conventional memory. In DOS 5 or later, the CONFIG.SYS DEVICEHIGH entry lets you install a device driver into reserved memory. Before you can use DEVICEHIGH, your CONFIG.SYS must contain entries similar to those shown here:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE  NOEMS
DOS=UMB
DEVICEHIGH=C:\DOS\ANSI.SYS
```

In this case, DOS will try to load the ANSI.SYS device driver discussed in Chapter 34 into reserved memory. The HIMEM.SYS driver provides extended memory support. The EMM386.EXE device driver provides access to the reserved memory. The NOEMS parameter tells DOS you don't want the driver to simulate expanded memory.

Do not try to install the HIMEM.SYS or EMM386 device drivers in reserved memory; these drivers must reside in conventional memory.

WARNING

The "Advanced Concepts" section of this chapter examines the EMM386.EXE device driver in detail. The CONFIG.SYS entries presented here are only for use on 386 and 486 systems that use only extended memory.

If you want DOS to also load itself into high memory, change the DOS entry to include HIGH, as shown here:

```
DOS=UMB,HIGH
```

If DOS does not have enough memory reserved to hold the driver software, DOS will load the driver in conventional memory as if you had used a DEVICE entry instead of DEVICEHIGH.

In Chapter 24 you learned that the CONFIG.SYS INSTALL entry lets you load four specific memory-resident DOS commands efficiently into memory. In DOS 5 or later, the LOADHIGH command lets you load memory-resident programs into reserved memory. LOADHIGH is not a CONFIG.SYS entry, it is an internal DOS command.

To determine the conventional memory usage and the amount of available upper memory, invoke MEM using the /CLASSIFY switch, shown here:

```
C:\> MEM  /CLASSIFY <ENTER>
Modules using memory below 1 MB:

  Name          Total      =   Conventional    +   Upper Memory
  --------  ----------------   ----------------   ---------------
  MSDOS       16685   (16K)      16685   (16K)         0    (0K)
  SETVER        624    (1K)        624    (1K)         0    (0K)
  HIMEM        1104    (1K)       1104    (1K)         0    (0K)
  EMM386          8080    (8K)         8080    (8K)          0     (0K)
  ANSI            4208    (4K)         4208    (4K)          0     (0K)
  SBPCD          10016   (10K)        10016   (10K)          0     (0K)
  RAMDRIVE        1200    (1K)         1200    (1K)          0     (0K)
  SMARTDRV       29296   (29K)        29296   (29K)          0     (0K)
  COMMAND         3168    (3K)         3168    (3K)          0     (0K)
  Free          581088  (567K)       581088  (567K)          0     (0K)

Memory Summary:

  Type of Memory          Size       =        Used        +        Free
  ----------------  ----------------    ----------------  ---------------
  Conventional         655360    (640K)      74272     (73K)  581088    (567K)
  Upper                     0      (0K)          0      (0K)       0      (0K)
  Adapter RAM/ROM      393216    (384K)     393216    (384K)       0      (0K)
  Extended (XMS)      3145728   (3072K)    2613248   (2552K)  532480    (520K)
  ----------------  ----------------    ----------------  ---------------
  Total memory        4194304   (4096K)    3080736   (3009K) 1113568   (1087K)
  Total under 1 MB     655360    (640K)      74272     (73K)  581088    (567K)
  EMS is active.
  Largest executable program size           580896    (567K)
  Largest free upper memory block                0      (0K)
  MS-DOS is resident in the high memory area.
```

Before you can use LOADHIGH, your CONFIG.SYS file must contain a DOS=UMB entry just as if you were using DEVICEHIGH. The following LOADHIGH command, for example, installs the FASTOPEN memory-resident software into reserved memory. If insufficient reserved memory is available, LOADHIGH will load the command into conventional memory.

```
C:\> LOADHIGH  FASTOPEN  C:=50 <ENTER>
```

DOS lets you abbreviate LOADHIGH as LH. If your AUTOEXEC.BAT loads memory-resident software, you may be able to load the software into reserved memory using LOADHIGH.

As you have learned, use of DEVICEHIGH and LOADHIGH to load device drivers and memory-resident programs into reserved memory frees memory for your programs. In some cases, a program's initial starting address may reside below 64Kb.

Some programs cannot execute below 64Kb and will display the "Packed file corrupt" error message when you execute them in this low memory region. If this error message occurs, use DOS 5 or later LOADFIX to direct DOS to load the program above 64Kb.

The format of the LOADFIX command is

```
LOADFIX ProgramName [parameters]
```

where *ProgramName* specifies the program you are trying to run, and *parameters* specifies the program's optional command-line parameters. You will not need to use LOADFIX unless you experience the "Packed file corrupt" error message.

Using the DOS 6 MEMMAKER Command

Although the DOS memory-manipulation commands and CONFIG.SYS entries are very powerful, they are often difficult concepts for many users to master. Therefore, DOS 6 provides the MEMMAKER command, which examines your system configuration and modifies your CONFIG.SYS, AUTOEXEC.BAT, and possibly your Windows SYSTEM.INI file. In other words, MEMMAKER automatically configures your system for optimal memory use. The format of the MEMMAKER command is as follows:

```
MEMMAKER [/BATCH] [/SWAP:Drive] [/UNDO] [/W:Buffer1,Buffer2] [/T]
```

The /BATCH switch directs MEMMAKER to run in a noninteractive (or batch) mode, in which MEMMAKER makes its changes without displaying user prompts. The /SWAP switch specifies the disk from which your system originally booted. If, after MEMMAKER runs, your system experiences errors, you can use the /UNDO switch to direct MEMMAKER to remove its changes. The /W switch lets you specify the size of the two translation buffers Windows uses for device I/O when running in protected mode. The default size is 12Kb. If you don't use Windows, set the buffer size to 0. Finally, the /T switch directs MEMMAKER to disable its IBM Token Ring network detection.

If you understand the DOS memory-manipulation facilities, you don't need to use the MEMMAKER command. However, if you do not yet feel comfortable with the memory-management settings, MEMMAKER can make the correct settings for you automatically.

Before you begin, make sure you have a bootable floppy disk, just in case something goes wrong. Also, you should print a copy of your current CONFIG.SYS and AUTOEXEC.BAT files. If MEMMAKER changes these files, MEMMAKER will store the original settings in files with the UMB extension (CONFIG.UMB and AUTOEXEC.UMB). To begin, start MEMMAKER as follows:

```
C:\> MEMMAKER <ENTER>
```

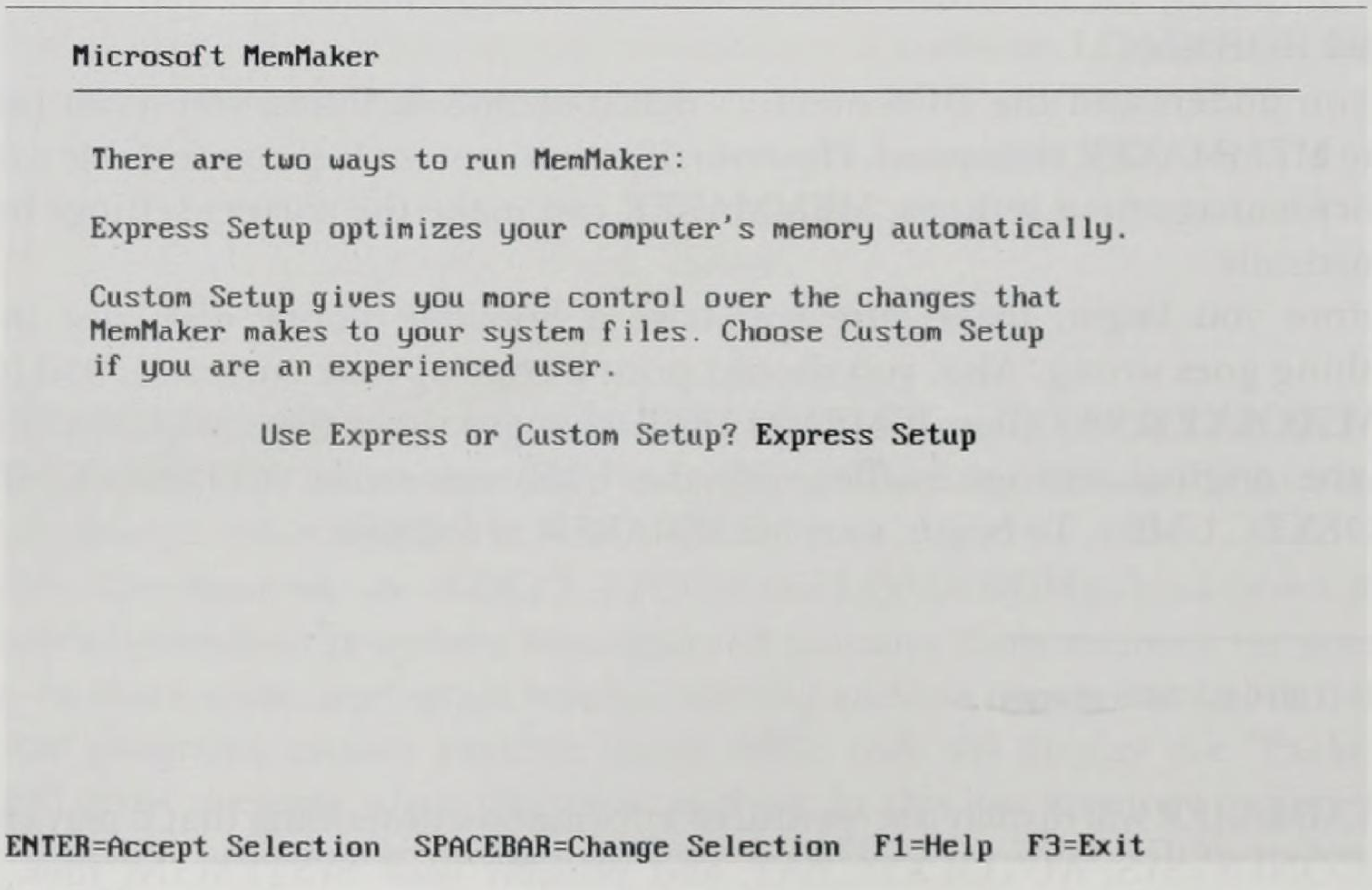
MEMMAKER will display a screenful of information describing that it may change your CONFIG.SYS, AUTOEXEC.BAT, and possibly your SYSTEM.INI files. Press ENTER to continue. MEMMAKER will display the screen shown in Figure 32-10 asking you if you want to perform a Custom or Express Setup.

Most novice users will select the Express Setup option, allowing MEMMAKER to make its own decisions. If you instead select Custom, MEMMAKER will display its Advanced Options as shown in Figure 32-11.

To change an option's setting, highlight the option using your keyboard arrow keys and press the spacebar. When you press ENTER, MEMMAKER will begin its configuration by examining your CONFIG.SYS and AUTOEXEC.BAT files. MEMMAKER will then display a screen telling you that your system will restart. Press ENTER and MEMMAKER will restart your computer. After your computer starts, MEMMAKER will automatically become active and will examine your system's memory usage, possibly making a few adjustments. Eventually, MEMMAKER will display a message telling you that your system will restart. Press ENTER. As your system starts, watch your screen closely for possible error messages. If no messages appear, MEMMAKER's configuration was probably successful. MEMMAKER will display one or more summary screens describing the operations that it has performed and the amount of memory you have gained. Press ENTER to continue. After your system starts, print a copy of your CONFIG.SYS and AUTOEXEC.BAT files and note the changes MEMMAKER has made.

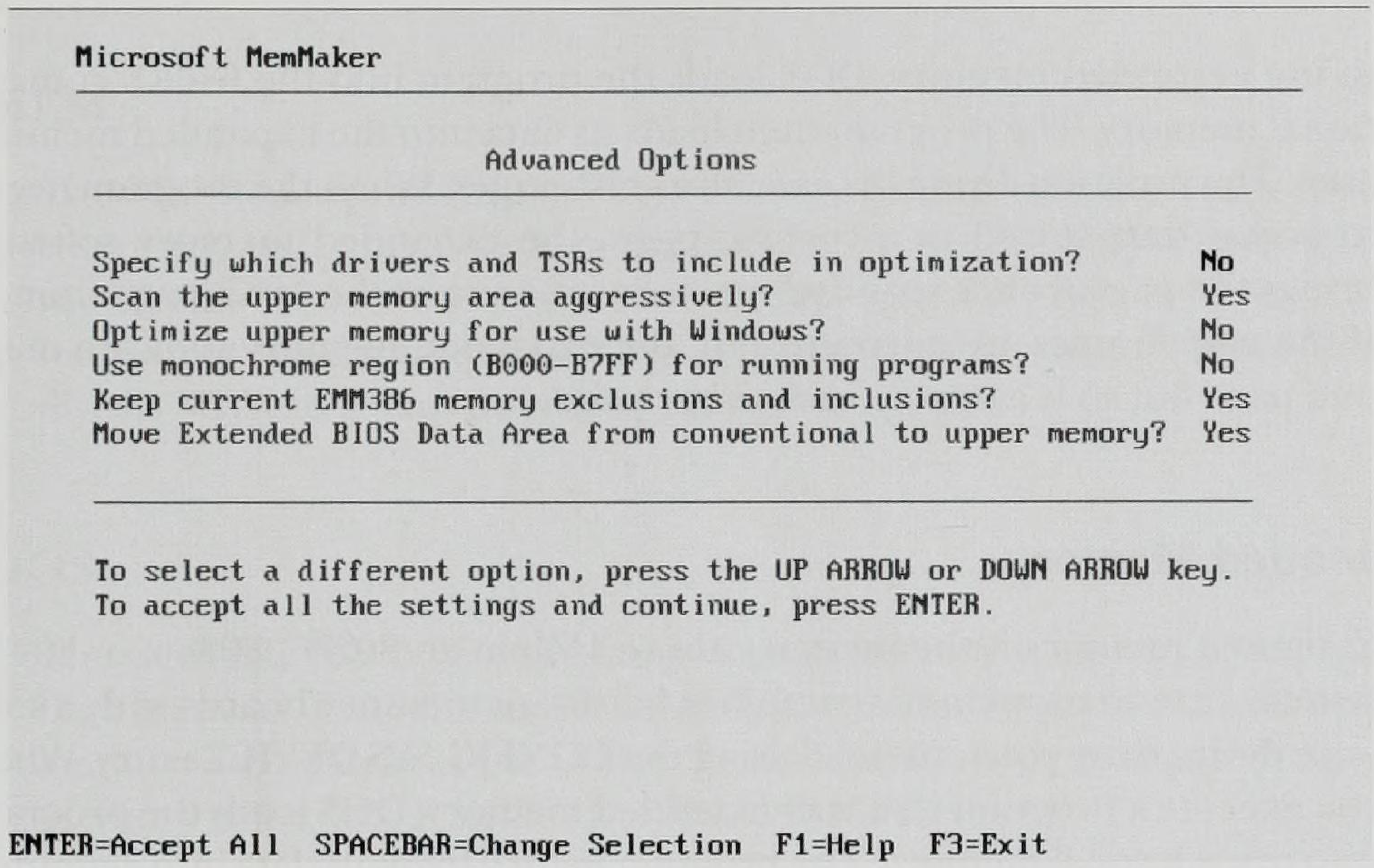
If an error message occurred during your system startup, edit your CONFIG.SYS and AUTOEXEC.BAT files and attempt to correct the entries causing the error. If you are unable to correct the error, you can use MEMMAKER's /UNDO switch as follows to remove MEMMAKER's changes.

```
C:\> MEMMAKER /UNDO <ENTER>
```



```
Microsoft MemMaker

There are two ways to run MemMaker:

Express Setup optimizes your computer's memory automatically.

Custom Setup gives you more control over the changes that
MemMaker makes to your system files. Choose Custom Setup
if you are an experienced user.

          Use Express or Custom Setup? Express Setup




ENTER=Accept Selection  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

Figure 32-10. *MEMMAKER's prompt for a Custom or Express Setup*



```
Microsoft MemMaker

                         Advanced Options

  Specify which drivers and TSRs to include in optimization?      No
  Scan the upper memory area aggressively?                        Yes
  Optimize upper memory for use with Windows?                     No
  Use monochrome region (B000-B7FF) for running programs?         No
  Keep current EMM386 memory exclusions and inclusions?           Yes
  Move Extended BIOS Data Area from conventional to upper memory? Yes

  To select a different option, press the UP ARROW or DOWN ARROW key.
  To accept all the settings and continue, press ENTER.

ENTER=Accept All  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

Figure 32-11. *MEMMAKER's Advanced Options*

SUMMARY

Conventional Memory

Conventional memory is your computer's memory from 0 to 640Kb. When your computer starts, DOS loads itself into the lower portion of conventional memory. If you invoke memory-resident programs such as PRINT, DOSKEY, or a network driver, DOS loads the program into the next available location in conventional memory.

Expanded Memory

Expanded memory lets the original IBM PC (8088) break the 640Kb boundary. Expanded memory combines hardware memory boards and powerful software.

Expanded memory uses a 64Kb region immediately above conventional memory as a buffer for four 16Kb pages. When you run a program written

SUMMARY

to use expanded memory, DOS loads the program into the 640Kb conventional memory. The program then loads its data into the expanded memory area. The program divides its data into 16Kb pages. When the program needs to access data stored in a specific page, the expanded memory software brings the page from expanded memory into one of the 16Kb page frames. If the page frames are currently full, the expanded memory software maps one page out so it can map in another page.

Extended Memory

Extended memory is the memory above 1Mb in an 80286, 80386, or 80486 system. Extended memory combines hardware memory boards with a software device drive you must load using the CONFIG.SYS DEVICE entry. When you execute a program that uses extended memory, DOS loads the program into conventional memory. The program in turn loads its data into extended memory. When the program needs to use data stored in extended memory, the extended memory software lets the program access the data directly without the time-consuming mapping to a page buffer that occurs with expanded memory. As a result, extended memory is much faster than expanded.

RAMDRIVE.SYS

The RAMDRIVE.SYS device driver lets you install a RAM drive in your computer's fast electronic memory. After you create a RAM drive, you access the drive using the drive letter RAMDRIVE.SYS displays on your screen, which is typically D or E. Because of their tremendous speed, RAM drives are very convenient for storing temporary files. Because RAM drives reside in your computer's electronic memory, the files on the drive are lost when you turn off your computer's power or restart DOS.

SMARTDRV.SYS

The SMARTDRV.SYS device driver installs a disk cache into extended or expanded memory. A disk cache works as a high-performance, large disk buffer. Because the disk cache resides in either extended or expanded memory, the cache does not consume conventional memory that DOS can use to run larger programs.

SUMMARY

MEM

The MEM command displays information about your use of conventional, extended, and expanded memory. If you include the /DEBUG switch, MEM will display the location and size of each program in memory as well as the location of standard device drivers. In DOS 5 or later, the /CLASSIFY switch directs MEM to group programs by memory usage.

DOS=

The CONFIG.SYS DOS entry lets you direct DOS 5 or later to install itself in high memory. In addition, the DOS= entry lets you request a link to reserved memory DOS can later use to install a device driver or command in reserved memory using DEVICEHIGH or LOADHIGH.

DEVICEHIGH=

The CONFIG.SYS DEVICEHIGH entry directs DOS 5 or later to install a device driver into reserved memory, making more conventional memory available to DOS to use for other programs. To use the DEVICEHIGH entry, your CONFIG.SYS file must contain a DOS=UMB entry, and the EMM386.EXE driver must be installed. The following DEVICEHIGH entry installs the RAMDRIVE.SYS driver in reserved memory:

```
DEVICEHIGH=C: \DOS\RAMDRIVE.SYS
```

If DOS does not have enough reserved memory to hold the driver, DOS will load the driver into conventional memory, just as if you had used the DEVICE entry.

LOADHIGH

The LOADHIGH command lets you load memory-resident software into the reserved area of memory above conventional memory and below 1Mb. Loading memory-resident software into reserved memory frees conventional memory for other programs. To use LOADHIGH, you must be using a 386 or 486, with the DOS=UMB entry in CONFIG.SYS and the EMM386.EXE device driver installed. If your computer does not have enough reserved memory available for the program, LOADHIGH will load the program into conventional memory. The format of LOADHIGH is

```
LOADHIGH CommandName [parameters]
```

SUMMARY

LOADFIX

If you load DOS into high memory and load your device drivers and memory-resident programs into reserved memory, you may actually free up so much conventional memory that a program's starting address is actually below 64Kb. Unfortunately, some programs cannot execute in this address range and will display the "Packed file corrupt" message when you attempt to execute them. Using the DOS 5 or later LOADFIX command, you can instruct DOS to load the program into memory above the 64Kb address, resolving the problem.

Hands On

Because the DOS memory-management capabilities differ greatly between DOS 5 or later and other versions of DOS, this section is divided into two parts: one part for DOS 5 or later users and one part for DOS users of any other version.

DOS 5 or later Users

To begin, invoke the MEM command to determine your system's current conventional, expanded, and extended memory usage.

```
C:\> MEM <ENTER>
```

Next, redirect MEM's output to your printer and print a copy of your CONFIG.SYS file, as shown here:

```
C:\> MEM  >  PRN <ENTER>

C:\> PRINT  \CONFIG.SYS <ENTER>
```

If you are using an 80286, 386, or 486 computer that has extended memory, make sure you install the HIMEM.SYS device driver in CONFIG.SYS.

```
DEVICE=C:\DOS\HIMEM.SYS
```

Next, if you are not using an expanded memory driver other than EMM386.EXE, place one of the following DEVICE entries in CONFIG.SYS to enable access to reserved memory:

```
DEVICE=C:\DOS\EMM386.EXE  NOEMS
```

or

```
DEVICE=C:\DOS\EMM386.EXE  RAM
```

Use the NOEMS entry if you want access to the reserved memory, but don't want DOS to simulate expanded memory using a portion of your computer's extended memory. Use the RAM entry if you want access to reserved memory and you want to simulate expanded memory.

Next, if you are using extended memory, use the following entry to load DOS into high memory and to establish links to reserved memory:

```
DOS=UMB,HIGH
```

Next, if your CONFIG.SYS file uses DEVICE to load other device drivers, use the DEVICEHIGH entry to load the drivers into reserved memory.

Likewise, if your AUTOEXEC.BAT installs memory-resident commands, use LOADHIGH to load the commands into reserved memory.

Using the CTRL-ALT-DEL key combination, restart your system. When the system restarts, again redirect MEM's output to your printer and print the contents of CONFIG.SYS.

```
C:\> MEM  >  PRN <ENTER>

C:\> PRINT  \CONFIG.SYS  <ENTER>
```

Compare your new memory usage using high and reserved memory to the original memory usage. Your new configuration should have more conventional memory available.

Use the following MEM command to display your computer's memory usage by program:

```
C:\> MEM /DEBUG  |  MORE <ENTER>
```

If you're using DOS 6, invoke MEM using the /DEBUG and /PAGE switches to display MEM's output a page at a time as shown in the following example.

```
C:\> MEM /DEBUG /PAGE <ENTER>
```

Extended or Expanded Memory DOS Users

If you are not using DOS 5 or later, but you are using extended or expanded memory, you can improve your system performance by installing disk-caching software. Like DOS 5 or later, Microsoft Windows provides the device driver SMARTDRV.SYS that installs a disk cache. If you are using extended memory, place a DEVICE entry similar to the following after the DEVICE entry that installs your extended memory driver:

```
DEVICE=C:\DOS\SMARTDRV.SYS
```

If the SMARTDRV.SYS device driver resides in your WINDOWS directory, use the following entry:

```
DEVICE=C:\WINDOWS\SMARTDRV.SYS
```

If you are using expanded memory, include the /A switch. You should note that the previous SMARTDRV.SYS entries install a disk cache using the default size of 256Kb. Depending on your available memory, you may want to create a larger or smaller disk cache.

If you install a disk cache, reduce the number of DOS disk buffers in CONFIG.SYS to three, as shown here:

```
BUFFERS=3
```

All DOS Users

Regardless of your DOS version or memory types, you may want to install a RAM drive using the RAMDRIVE.SYS device driver. If you are using expanded or extended memory, you can install the RAM drive outside of conventional memory using the /A or /E switches. If you don't specify one of these two switches, DOS will create the RAM drive in conventional memory.

The following CONFIG.SYS entry creates a 64Kb RAM drive in conventional memory:

```
DEVICE=C:\DOS\RAMDRIVE.SYS
```

NOTE *If you are using DOS 5 or later, create a RAM drive and assign the TEMP environment entry to point at the drive. DOS will use the fast RAM drive for its temporary files, which will improve the performance of your I/O redirection operations.*

Review

1. What is conventional memory?
2. What is expanded memory?
3. What is extended memory?
4. What is reserved memory?
5. What is high memory?

Answers

1. Conventional memory is your computer's memory from 0 to 640Kb. All computers use conventional memory. Most DOS programs run in conventional memory.
2. Expanded memory uses hardware and software to break the 640Kb boundary of the IBM PC (8088). The expanded memory software reserves a 64Kb region of the computer's reserved memory (see Question 4) for use as a page frame. DOS will load programs into conventional memory as always. The program itself can then access up to 16Mb of data stored in expanded memory. The expanded memory software maps references to a program's data (the address of the data) from the memory region called a *bank*, which resides on the expanded memory board, into one of four page frames in reserved memory. If all four page frames are in use, the expanded memory software swaps the contents of one page frame back to the memory region above 1Mb to make room for the new page. Because of this continual swapping of pages, expanded memory is slower than conventional memory.
3. Extended memory is the memory above 1Mb available in the IBM PC AT (80286), 386, and 486 computers. To use extended memory, you must install an extended memory manager (device driver). The PC AT supports up to 16Mb of extended memory. The 386 and 486 support up to 4 gigabytes!
4. Reserved memory is the 384Kb area of memory above 640Kb and below 1Mb. The IBM PC uses a portion of reserved memory for video output and memory-mapped device I/O. In DOS 5 or later, using the CONFIG.SYS DEVICEHIGH entry and LOADHIGH command, you can install device drivers and commands into reserved memory, freeing up conventional memory for use by DOS.

5. High memory is the first 64Kb memory region above 1Mb. In DOS 5 or later, you can use the CONFIG.SYS DOS=HIGH entry to load DOS into high memory, freeing up conventional memory for program use.

Advanced Concepts

The previous discussion of HIMEM.SYS and EMM386.EXE presented only the driver's basic capabilities. This section presents each of the driver's parameters and switches.

HIMEM.SYS

HIMEM.SYS allows your programs to use extended memory. You must install HIMEM.SYS using a DEVICE entry in your CONFIG.SYS file before any other drivers that use extended memory, such as RAMDRIVE.SYS, EMM386.EXE, or SMARTDRV.SYS.

The complete format of HIMEM.SYS is

```
DEVICE=C:\DOS\HIMEM.SYS  [/HMAMIN=MinimumMemory]
      [/NUMHANDLES=HandleCount] [/INT15=Kb] [/MACHINE:nnnn]
      [/A20CONTROL:ON|OFF] [/SHADOWRAM:ON|OFF] [/CPU CLOCK:ON|OFF]
```

The /HMAMIN switch specifies the minimum amount of high memory in kilobytes a program must use before the program can use the high memory area. The default value is 0. The range of valid values is 0 through 63. By default HIMEM.SYS gives the high memory area to the first program that requests it. Using /HMAMIN, you may be able to reserve the high memory area for the desired program.

Programs allocate extended memory in variable-sized blocks. When a program allocates a block, HIMEM.SYS returns a unique value called a *handle* that the program must use each time it references the block. The /NUMHANDLES switch specifies the number of available handles from 1 to 128. The default value is 32. Each handle requires six bytes of conventional memory.

Programs can access extended memory using hardware interrupt 15H. The /INT15 switch specifies the size in Kb of a memory buffer HIMEM.SYS provides for the INT 15H interface. The default size is 0. Valid values range from 0 to 64Kb. The ROM-BIOS lets programs access extended memory using hardware interrupt 15. The extended memory BIOS interface occurs outside of the extended memory-management software. You only need to use /INT15 if a program uses the BIOS to access extended memory in this way.

To access the high memory area, you computer uses special software called the A20 handler (A stands for address), which handles memory address references that use address line 20 (a memory reference above 1Mb). In most cases, HIMEM.SYS can correctly determine the computer's A20 value. If you are using one of the machines listed in Table 32-1, and HIMEM.SYS displays A20 error messages when you install it, include the /MACHINE switch to assist HIMEM.SYS.

The /A20CONTROL switch specifies whether or not HIMEM.SYS takes control of the computer's A20 handler. If you specify ON (the default setting), HIMEM.SYS will take control of the A20 line, if the computer has it on or off. If you specify OFF, HIMEM.SYS only takes control of the A20 line if it is initially off.

Your computer's BIOS is stored in read-only memory whose contents never change, even when your computer's power is off. Compared to RAM, this read-only memory, or ROM, is slow. Many computers will copy the BIOS from slower ROM to locations in the faster RAM. The /SHADOWRAM switch directs HIMEM.SYS to disable the use of shadow RAM to regain the memory for other uses. The ON value enables shadow RAM, while OFF disables it. Not all computers support shadow RAM. If your computer has less than 2Mb of RAM, HIMEM.SYS uses OFF by default.

Code	Number	Machine
AT	1	IBM PC AT
PS2	2	IBM PS/2
PT1Cascade	3	Phoenix cascade BIOS
HPVECTRA	4	HP Vectra A and A+
ATT6300PLUS	5	AT&T 6300 Plus
ACER1100	6	Acer 1100
TOSHIBA	7	Toshiba 1600 and 1200XE
WYSE	8	Wyse 12.5 MHz 286
TULIP	9	Tulip SX
ZENITH	10	Zenith ZBIOS
AT1	11	IBM PC AT
AT2	12	IBM PC AT (alternative delay)
CSS	12	CSS Labs
AT3	13	IBM PC AT (alternative delay)
PHILIPS	13	Philips
FASTHP	14	HPVECTRA

Table 32-1. *HIMEM.SYS A20 Handler Values*

In some cases, your computer's speed may slow down when you install HIMEM.SYS. The /CPUCLOCK switch lets HIMEM.SYS control the speed of your computer's system clock. When you specify ON, HIMEM.SYS can control the clock speed. The default setting is OFF.

EMM386.EXE

Depending on your applications, you may have a program that supports expanded memory and not extended memory. Using the EMM386.EXE driver, you can direct DOS to simulate expanded memory with a portion of your computer's extended memory, letting the program run. To use EMM386.EXE, you must be using a 386 or 486 computer. You must also install HIMEM.SYS before you can use EMM386.EXE.

The complete form of EMM386.EXE is as follows:

```
DEVICE=C:\DOS\EMM386.EXE  [ON|OFF|AUTO] [Kb] [W=ON|OFF]
      [Mnn |FRAME=Segment |/PSegment] [PPage=Segment]
      [I=Range] [X=Range] [B=Segment] [L=MinimumExtendedMemory]
      [A=AlternateRegisters] [H=Handles] [D=DMAKb]
      [RAM] [NOEMS]
```

By default, EMM386.EXE uses the ON parameter, which keeps expanded memory support active. The AUTO parameter directs the driver to only provide expanded memory support when a program requests it. The OFF mode loads the driver but then suspends it (makes it inactive), letting a program enable the driver support at a later time.

The Kb parameter specifies the amount of expanded memory desired from 16Kb to 32,768Kb (32Mb). The EMM386.EXE driver rounds down the value specified to a multiple of 16Kb. The default is 256Kb.

The W parameter provides support for the Weitek math coprocessor, which uses a 64Kb-block memory-mapped address space. If your system does not have a math coprocessor, you can omit this parameter. The default setting for W is OFF (no support).

Expanded memory uses a 64Kb region of your computer's reserved memory called the *page frame.* Depending on your hardware configuration, you may need to specify the base address of the page frame. The EMM386.EXE driver gives you several ways to specify the base address. First the M*nn* parameter uses one of the values specified in Table 32-2.

The FRAME and /P parameters let you explicitly state the segment address for the page frame. Using these two parameters you can select one of the base addresses listed in Table 32-2.

Using M, FRAME, AND /P, the following entries specify the same base address:

```
M5   FRAME=D000   /PD000
```

Value	Base Address	Value	Base Address
1	C000H	8	DC00H
2	C400H	9	E000H
3	C800H	*10	8000H
4	CC00H	*11	8400H
5	D000H	*12	8800H
6	D400H	*13	8C00H
7	D800H	*14	9000H

Table 32-2. *Page-Frame-Use Address Values (*Values 10 through 14 are only for computers with 512Kb of RAM)*

If you use the M, FRAME, or /P parameters, you define the base address for all four page locations. Thus, EMM386.EXE allocates a contiguous 64Kb memory region.

Using the P parameter, you can specify individual locations for each page. As before, the addresses you must use are in Table 32-2. To maintain compatibility with LIMS 3.2, pages 0, 1, 2, and 3 must be in contiguous memory locations.

The I parameter specifies a range of segment address values to be used for an expanded memory page. By default, the expanded memory software tries to determine which memory locations it can safely use to store pages. If you are aware of a safe region, you can use the I parameter to tell the expanded memory software.

The X parameter specifies a range of segment addresses to be excluded from storing extended memory pages. The X parameter has precedence over I, should ranges overlap. The following entry excludes the range C400H to C800H:

```
X=C400-C800
```

The B parameter specifies the lowest segment address EMM386.EXE can use for a page address from 1000H through 4000H. The default value is 4000H.

The L parameter specifies the amount of extended memory in Kb that must be available after you load EMM386.EXE. The EMM386.EXE driver will determine your expanded memory use by subtracting the extended memory size specified from the available extended memory.

To improve your performance during task-switching operations in Windows or the DOS shell, you can specify the number of alternate register sets from 0 to 254. The default size is 7. Each alternate register set uses approximately 200 bytes of memory.

Like extended memory, when programs allocate blocks of expanded memory, the program receives a handle that uniquely identifies the block. By default, EMM386.EXE supports 64 handles. Using the H parameter, you can specify support for 2 through 255 handles.

Different block devices can use a special high-speed chip (instead of the CPU) to transfer data directly from the device into memory. This process of bypassing the CPU to transfer data into memory is *direct memory access* or *DMA*. To perform direct access, the DMA chip needs a buffer to store the data that typically resides in the reserved memory. The D parameter lets you specify the amount of memory in kilobytes EMM386.EXE must reserve for direct memory buffers from 16Kb through 256Kb. The default size is 16Kb.

The EMM386.EXE driver lets you simulate expanded memory, and lets DOS load memory-resident programs and device drivers into reserved memory. If you want to simulate expanded memory and access reserved memory, use the RAM parameter. If you want to access reserved memory without simulating expanded memory, use NOEMS.

The following examples illustrate different uses of EMM386.EXE. The first entry simulates 256Kb (the default) of expanded memory:

```
DEVICE=C:\DOS\EMM386.EXE
```

The following entry includes a size 2048 to simulate 2Mb of expanded memory:

```
DEVICE=C:\DOS\EMM386.EXE  2048
```

This entry directs EMM386.EXE to place the page frame at location D000H:

```
DEVICE=C:\DOS\EMM386.EXE  FRAME=D0000
```

Finally, this entry directs EMM386.EXE to simulate 512Kb of expanded memory and to provide access to reserved memory:

```
DEVICE=C:\DOS\EMM386.EXE   512   RAM
```

Key Terms

Conventional memory The computer's memory from 0 to 640Kb.

Expanded memory A memory-management technique that combines hardware and software to let the IBM PC (8088) break its 640Kb memory boundary. Expanded memory exchanges 16Kb blocks of memory from locations called banks that reside on the expanded memory board, into a page-frame buffer stored in reserved memory.

Extended memory The memory above 1Mb on an 80286, 386, or 486 computer.

Reserved memory The 384Kb memory region between 640Kb and 1Mb. The computer uses part of reserved memory for a video interface, part to communicate with other hardware devices, and, optionally, a 64Kb region for expanded memory. In DOS 5 or later, the DEVICEHIGH CONFIG.SYS entry and the LOADHIGH command let you install device drivers and memory-resident commands into unused portions of reserved memory.

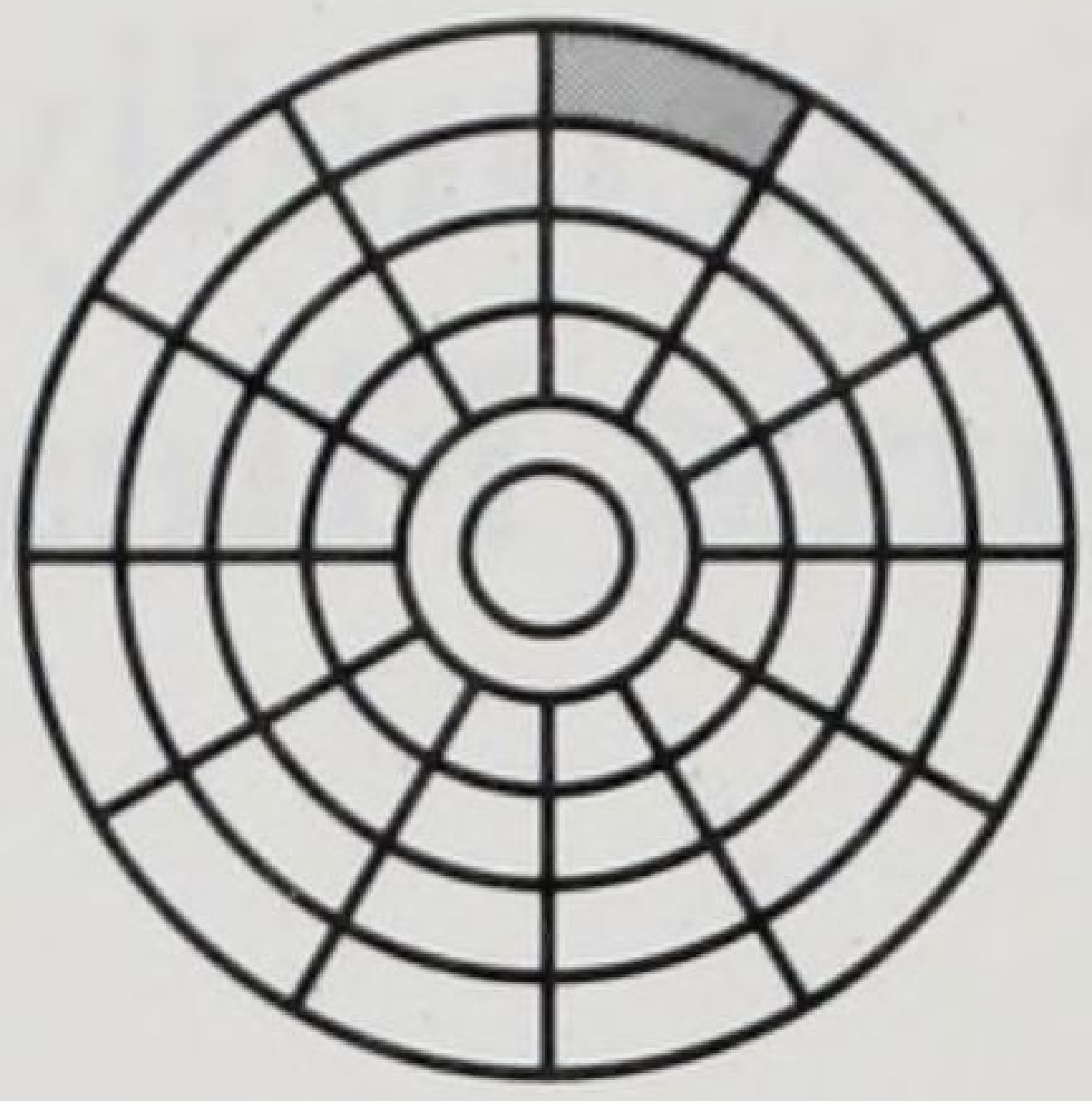
Chapter 33

How DOS Actually Stores Files

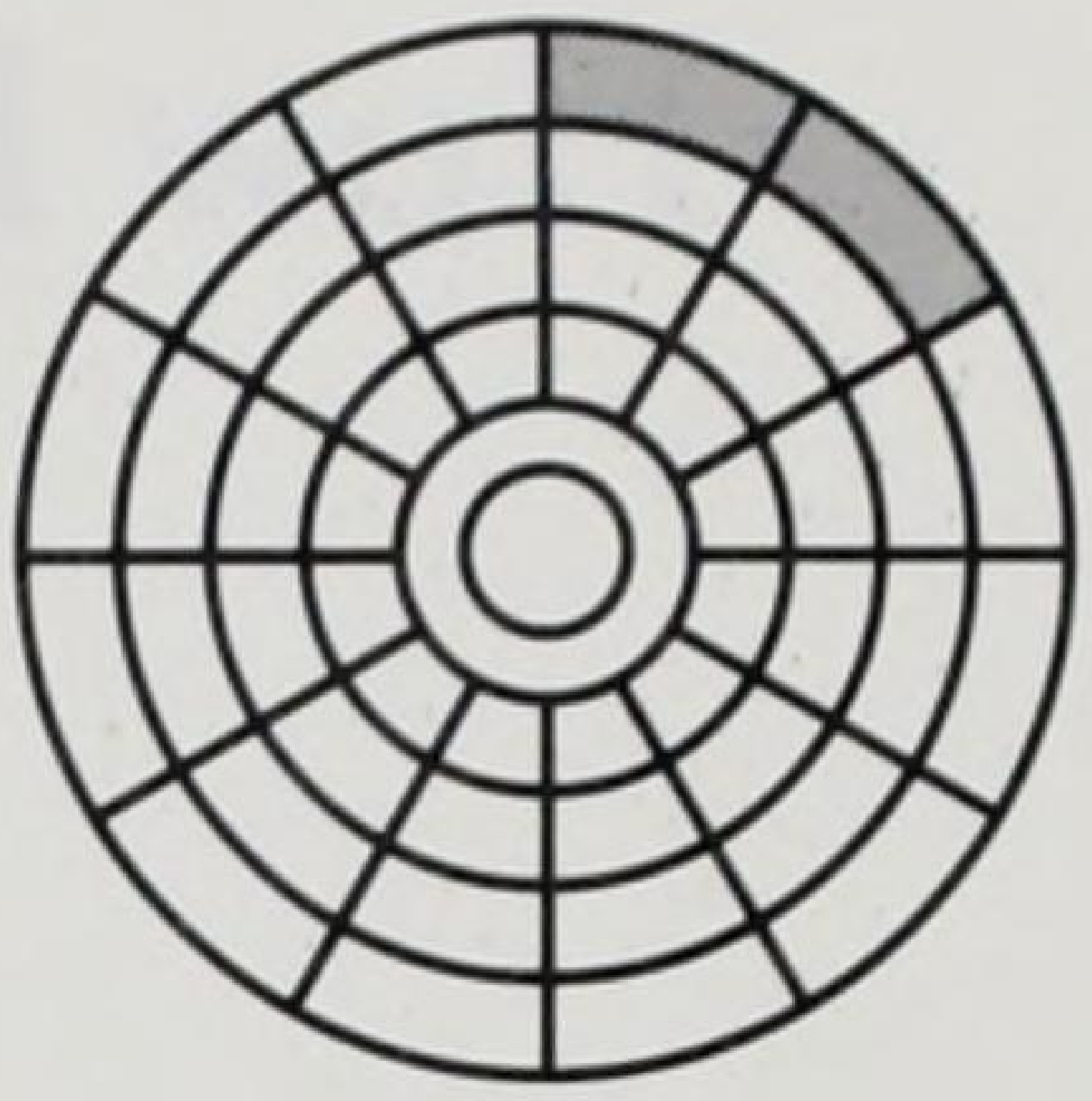
Throughout the previous chapters you have learned that DOS stores your files in disk sectors. For efficiency, DOS groups sectors to form allocation units (clusters). You have learned a directory is a list of filenames and related information such as file size, attributes (read-only, archive, hidden, and system), as well as the file's date and time stamps that tell you when the file was created or last changed. In addition, you have read brief discussions about the file allocation table (FAT) that serves as the road map DOS follows to locate the clusters belonging to each file. In this chapter you will see how all these concepts work together and will more fully understand the DOS file system. After you understand how DOS stores files, you can more efficiently use disk and file utilities to improve your system performance.

The File Allocation Table

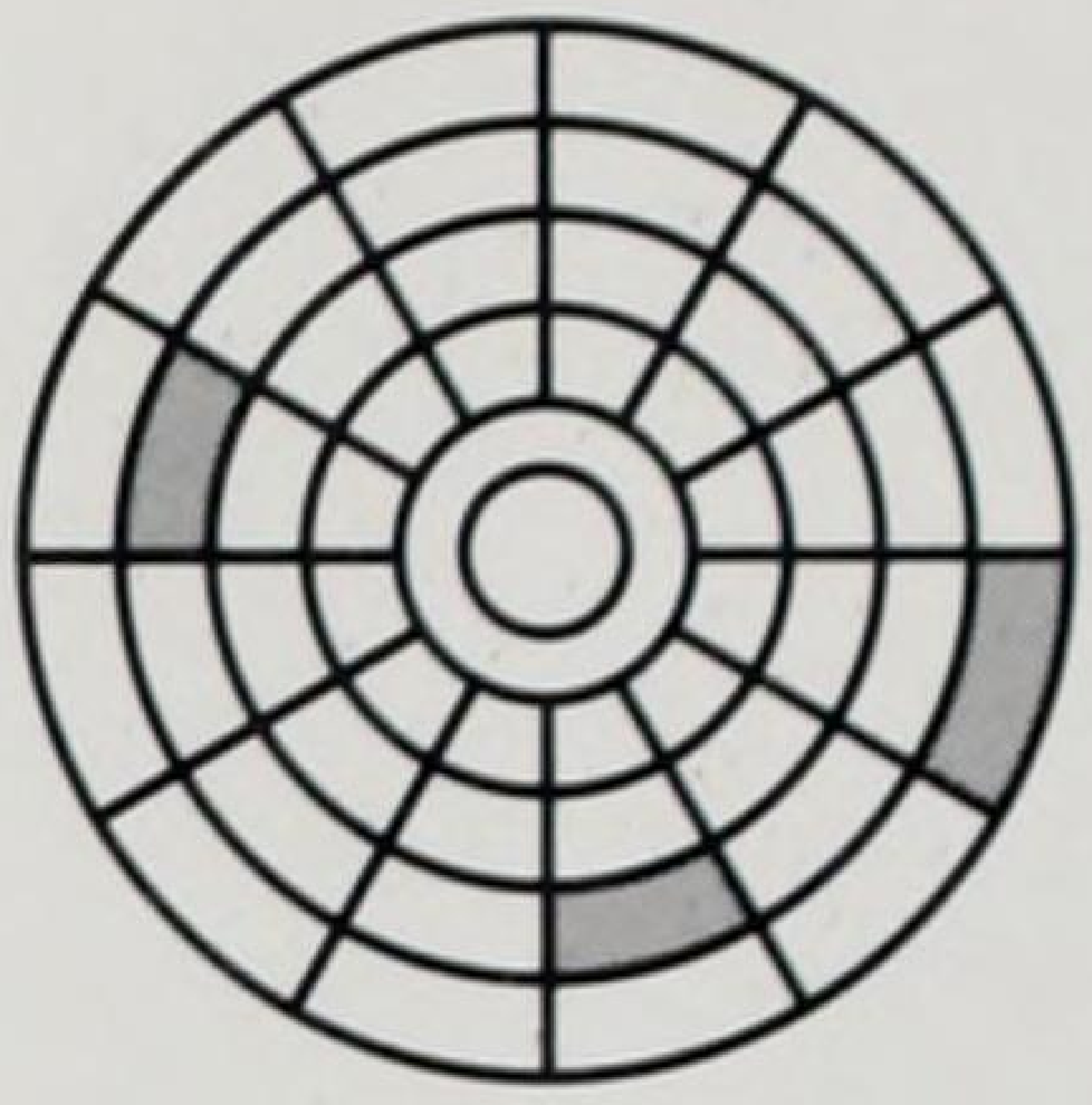
DOS stores your files on disks in groups of sectors called clusters or allocation units. Depending on the file's size, it may fit in a single cluster or may require many clusters. DOS tries to store a file's clusters in consecutive disk locations. In some cases, however, another file's cluster may prevent DOS from using consecutive cluster locations for a file, and the file becomes fragmented. The following illustration shows three files. The first file, ONE, fits in one cluster; the second, file TWO, resides in two consecutive clusters; while the third file, THREE, resides in three clusters dispersed across the disk. Each cluster may contain multiple disk sectors.



File ONE



File TWO



File THREE

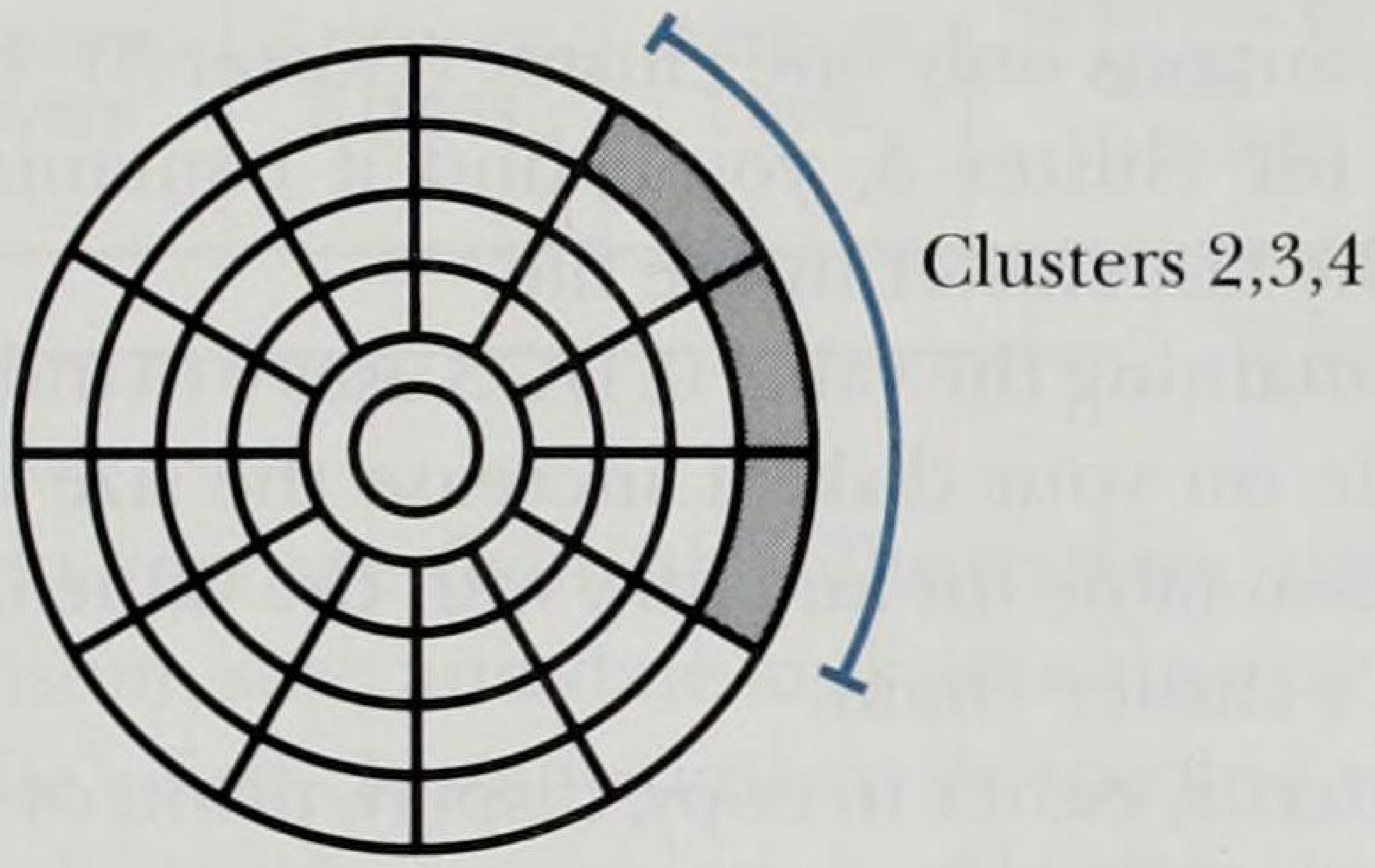
You learned when you examined CHKDSK in Chapter 25 that DOS divides your disk into a fixed number of clusters. Depending on your disk size, the number of clusters on the disk will differ. Table 33-1 lists the number of clusters available for several commonly used disk sizes. It should be noted that each cluster may contain multiple disk sectors.

As you have seen, DOS can store a file's contents on any unused cluster on your disk. To keep track of each file's clusters, DOS keeps a table on disk called the file allocation table. The file allocation table keeps track of every cluster on your disk, noting whether the cluster is in use, available for use, or has been marked unusable by FORMAT.

DOS uses the file allocation table to locate each file's clusters on disk, storing the list of cluster numbers that make up each file. Figure 33-1 illustrates a file that resides in clusters 2, 3, and 4. Using the file allocation table, DOS builds a chain of cluster numbers. If you look at the file allocation table entry for cluster 2, you'll find it contains the value 3, which corresponds to the next cluster in the file. Likewise, if you look at the entry for cluster 3, you'll find it contains the value 4, again the next cluster in the file. Cluster 4, in this case, is the file's last cluster. Within the file

Disk Type	Number of Clusters
360Kb	354
720Kb	713
1.2Mb	2371
1.44Mb	2847
2.88Mb	2863
10Mb	2549

Table 33-1. *The Number of Clusters Available on Commonly Used Disk Sizes*



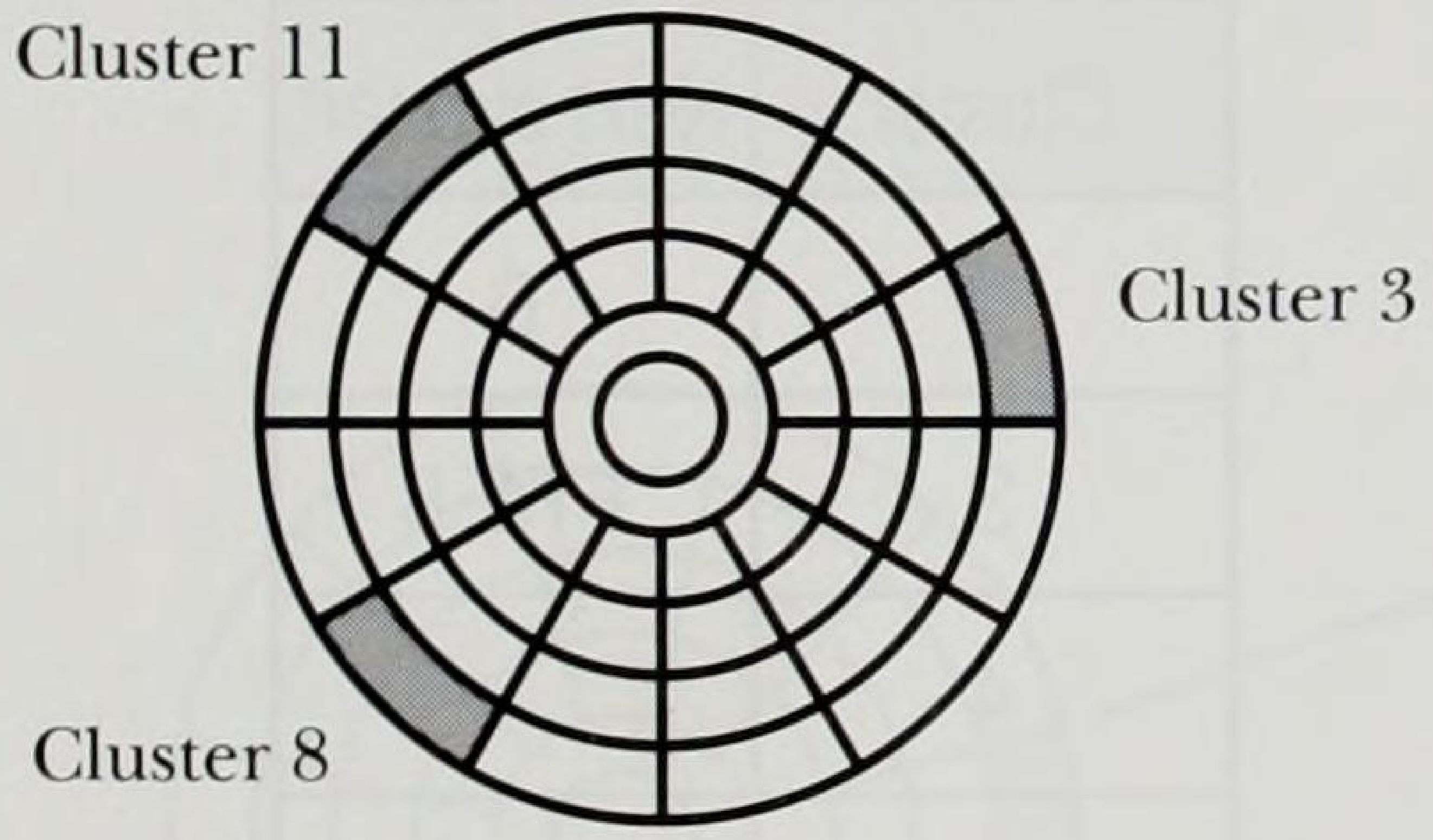
Cluster	Next cluster
2	3
3	4
4	FFF8
⋮	⋮
:	:

File allocation table

Figure 33-1. *A three-cluster file with its corresponding file allocation table entries*

allocation table, DOS uses the hexadecimal value FFF8 to represent the end of the file. You can see that the file resides in consecutive clusters.

Figure 33-2 illustrates a second file that again contains three clusters. In this case, however, the file is fragmented, with its data dispersed in clusters 3, 8, and 11. As



Cluster	Next cluster
3	8
⋮	⋮
8	11
:	:
11	FFF8

File allocation table

Figure 33-2. *A fragmented file with its corresponding file allocation table entries*

before, however, you can trace the file's cluster map beginning at cluster 3, to cluster 8, and to the file's last cluster in cluster 11.

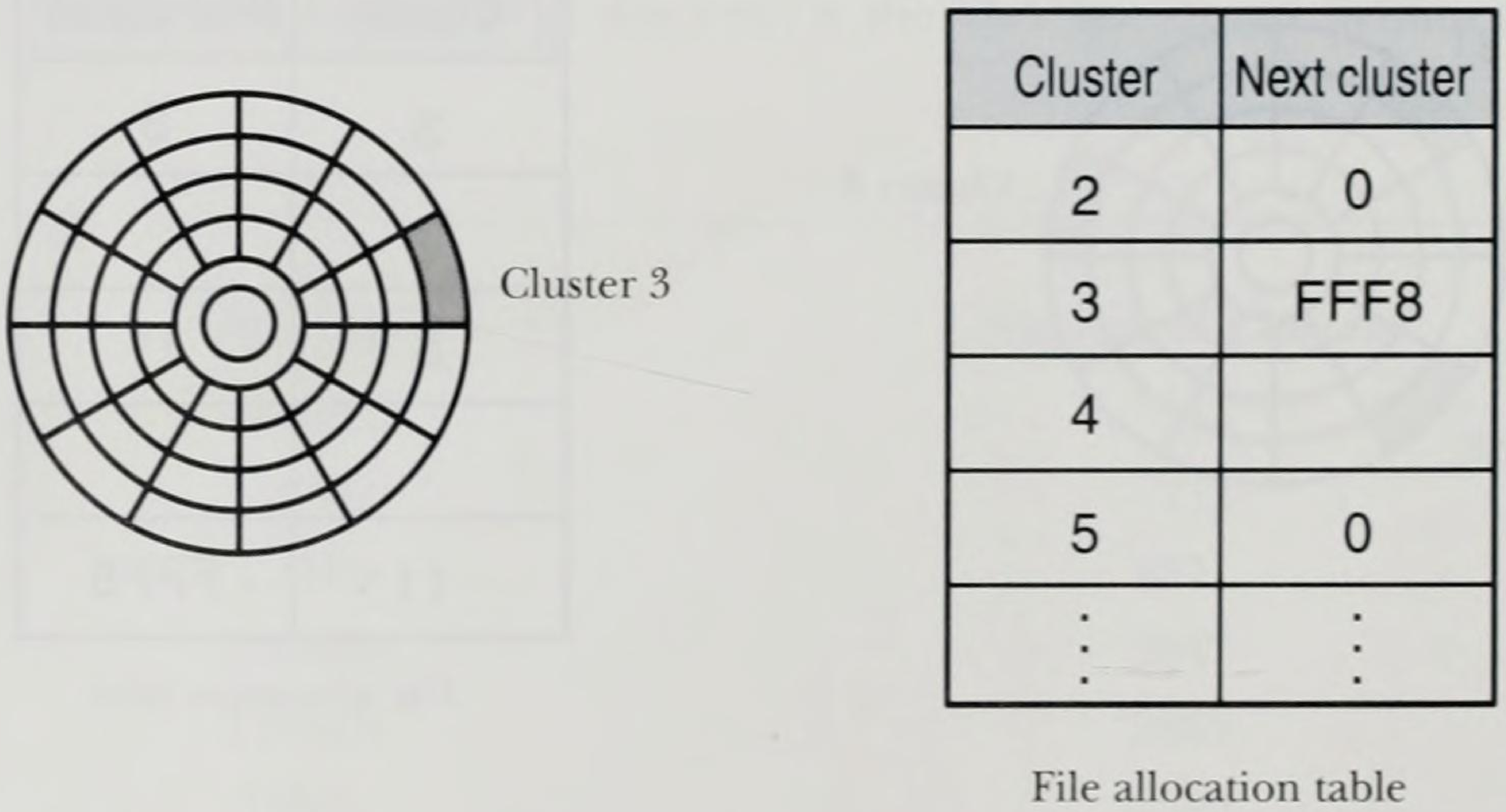
Last, Figure 33-3 illustrates a file that contains only one cluster (cluster 3). If you examine the file allocation table entry for cluster 3, you'll find it contains the hexadecimal value FFF8, indicating it is the last cluster in the file.

Note the file allocation table entries containing the value 0. DOS uses 0 to indicate an unused cluster. When you create a file on your disk or increase the size of an existing file, DOS searches the file allocation table for an entry with the value 0 and allocates the entry, placing it into the file's cluster chain.

When DOS needs to access a file's contents, either to copy, display, print, or even execute the file's contents, DOS follows the file's cluster chain in the same way you just have. In our case, however, the book told you each file's starting cluster. DOS, however, learns the file's starting cluster from the file's directory entry, as discussed in the next section.

Directory Entries

A directory is a list of filenames and key information about each file. Each time you create a file on disk, DOS creates a *directory entry* for the file. Each directory entry contains 32 bytes of information, as shown in the following:



Cluster	Next cluster
2	0
3	FFF8
4	
5	0
⋮	⋮

File allocation table

Figure 33-3. A file with only one cluster and its corresponding file allocation table entry

Filename 8 bytes	Extension 3 bytes	Attribute 1 byte	Reserved by DOS 10 bytes	Time 2 bytes	Date 2 bytes	Starting cluster 2 bytes	File size 4 bytes

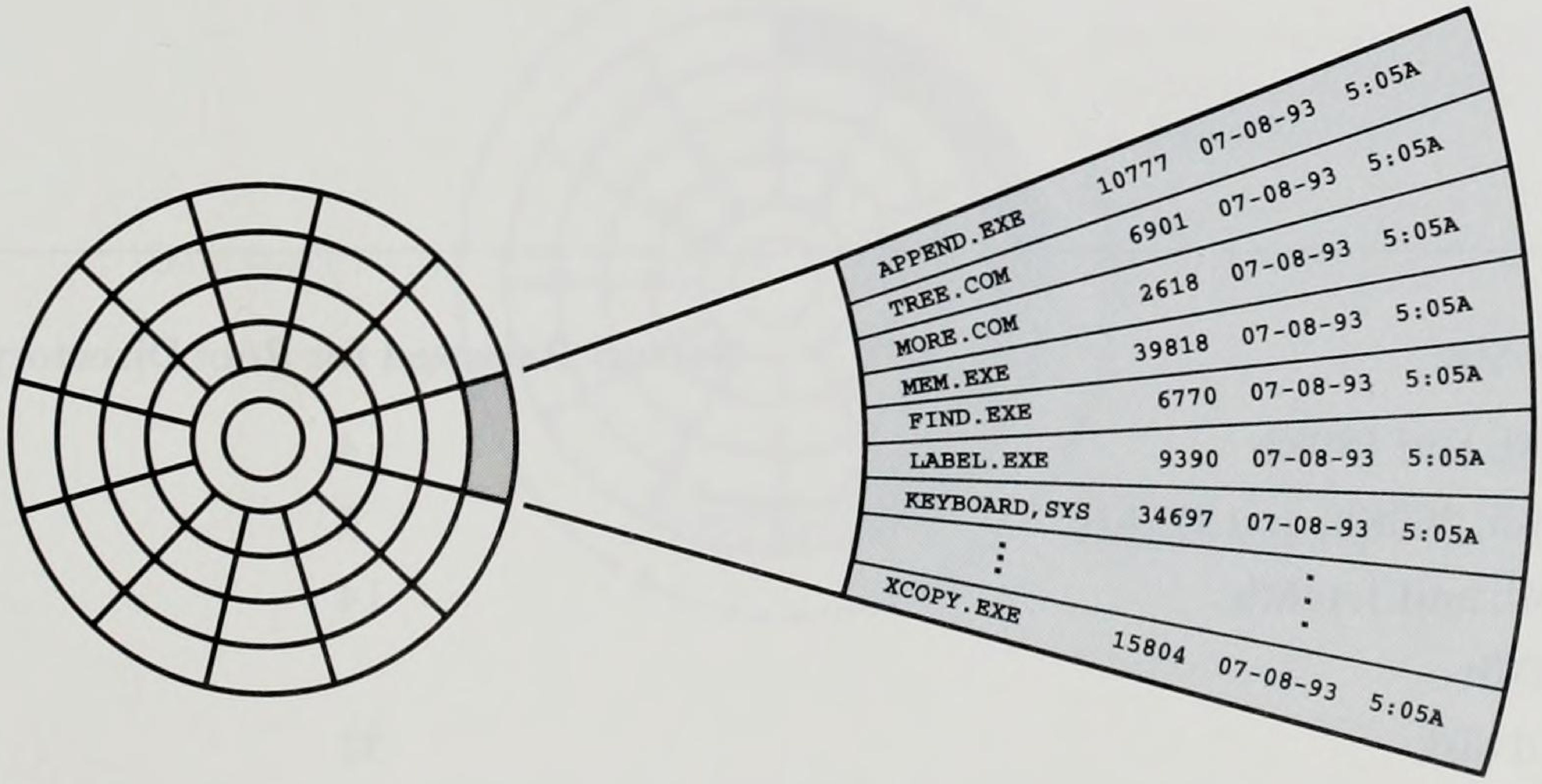
32 bytes

As you can see from the illustration, the directory entry stores the file's name, extension, size, and date and time stamps. In addition, the directory entry contains the file's first cluster number. By starting with this cluster value, DOS can locate the file's clusters by traversing the file allocation table.

DOS Stores Directories on Disk Just as Files

Each time you use MKDIR to create a directory, DOS creates a directory entry to store the directory name specified. To the user, the only difference between the directory and file entry is that directory entries appear in directory listings with the letters <DIR> instead of a file size. To distinguish a file entry from a directory entry, DOS assigns directory entries a unique attribute value, just as it would assign a read-only, hidden, system, or archive attribute.

When MKDIR creates a directory, DOS allocates a disk cluster to store the directory entries. As the number of files in your directory list grows, DOS allocates additional clusters to hold the entries, updating the directory's cluster chain in the file allocation table. Figure 33-4 illustrates how DOS might store a directory on disk.



```
APPEND.EXE     10777  07-08-93  5:05A
TREE.COM        6901  07-08-93  5:05A
MORE.COM        2618  07-08-93  5:05A
MEM.EXE        39818  07-08-93  5:05A
FIND.EXE        6770  07-08-93  5:05A
LABEL.EXE       9390  07-08-93  5:05A
KEYBOARD,SYS   34697  07-08-93  5:05A
   :                      :
XCOPY.EXE      15804  07-08-93  5:05A
```

Figure 33-4. *DOS stores directories on disk in one or more clusters, just as it stores files*

DOS Stores the FAT and Root Directory at Fixed Locations

The first sector on every disk contains the DOS boot record. If you have formatted the disk as a bootable system disk, the boot record contains a small program that begins the DOS startup process. If the disk is not bootable, the boot record contains a small program that causes your computer to display the following error message:

```
Non-System disk or disk error
Replace and strike any key when ready
```

In addition to containing this small program, the boot record contains values that describe your disk size, root directory size, and cluster size.

Beginning in the sector immediately following the boot record, DOS stores the file allocation table (FAT). As you can imagine, because the file allocation table contains the road map DOS uses to locate your files on disk, a disk error causing damage to such a sector could have a devastating impact. To reduce the chance of lost information due to a disk error in the FAT, DOS stores two copies of the FAT on your disk. Should a disk error ever render the first file allocation table unusable, DOS can still access your files using the second FAT copy.

Following your file allocation table, DOS reserves several sectors on your disk to store the root directory. The number of sectors DOS reserves depends on your disk size. Table 33-2 defines the number of sectors DOS reserves for common disk sizes.

Knowing each disk sector contains 512 bytes, and each directory entry requires 32 bytes, you can calculate the maximum number of root directory entries using the following formula:

Disk Size	Sectors Reserved for Root Directory
160Kb and 180Kb	4
320Kb, 360Kb, and 720Kb	7
1.2Mb and 1.44Mb	14
2.88Mb	
Hard disk	32

Table 33-2. Number of Sectors Reserved for Root Directory Entries

$$\text{Maximum root directory entries} = \frac{(\text{Root directory sectors}) \times (\text{Sector size})}{32 \text{ bytes per entry}}$$

The root directory is the only directory on your disk whose size is fixed. The disk space that follows the root directory is called *data space.* It is the disk space available for use by your files. Figure 33-5 illustrates the DOS disk layout.

How FORMAT Works

You learned in Chapter 7 that FORMAT prepares a disk for use by DOS. As you know, FORMAT examines your disk for bad sectors and marks the bad sectors as damaged, preventing their use by DOS. To mark a sector as unusable, FORMAT places a special value in the file allocation table entry for the cluster associated with the damaged sector. For hard disks, FORMAT uses the hexadecimal value FFF7. Figure 33-6 illustrates a disk with damaged sectors in clusters 3 and 7. Note the file allocation table values for those entries. When DOS is looking for disk space to store a file, DOS knows not to use clusters containing this value.

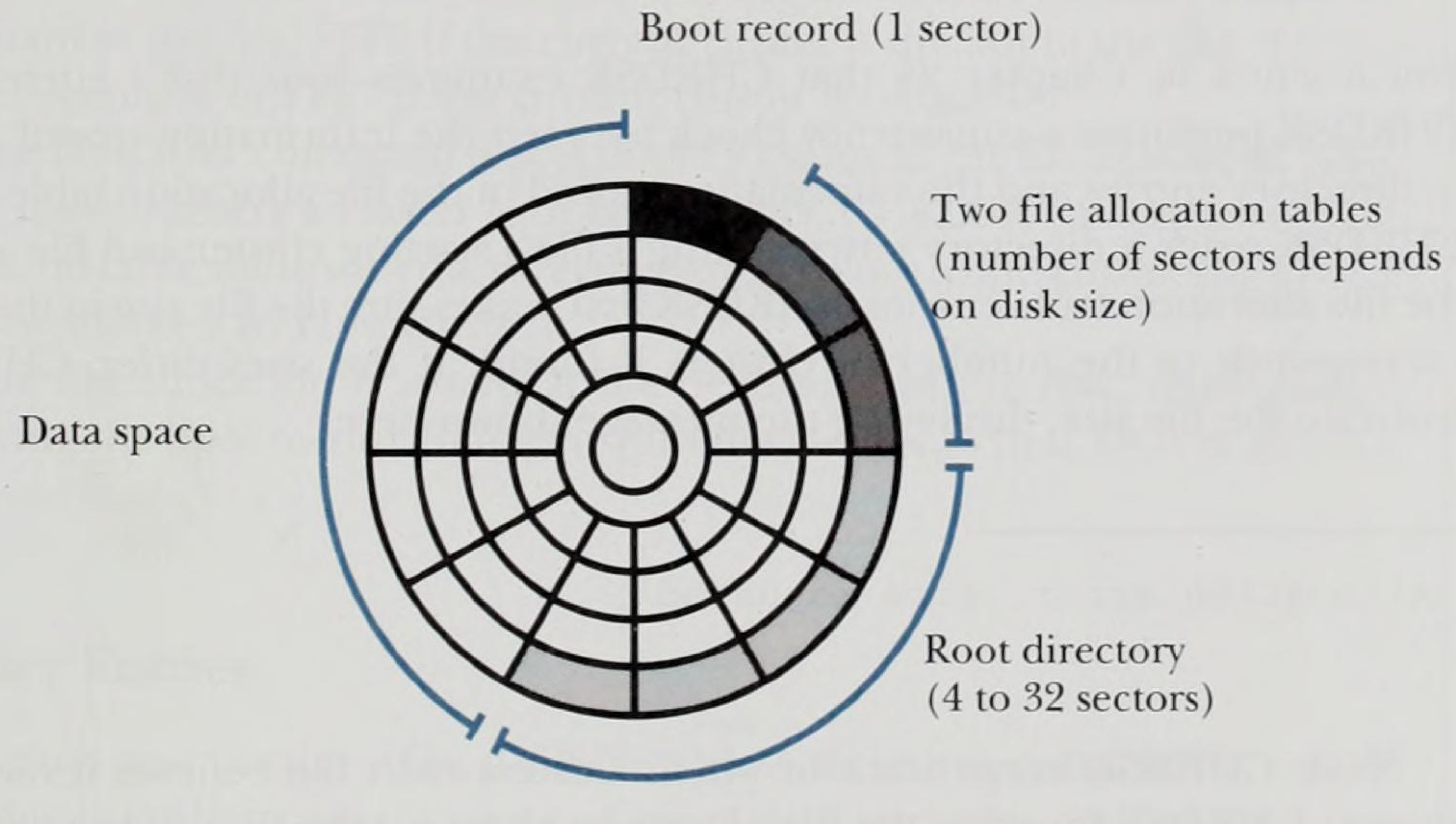
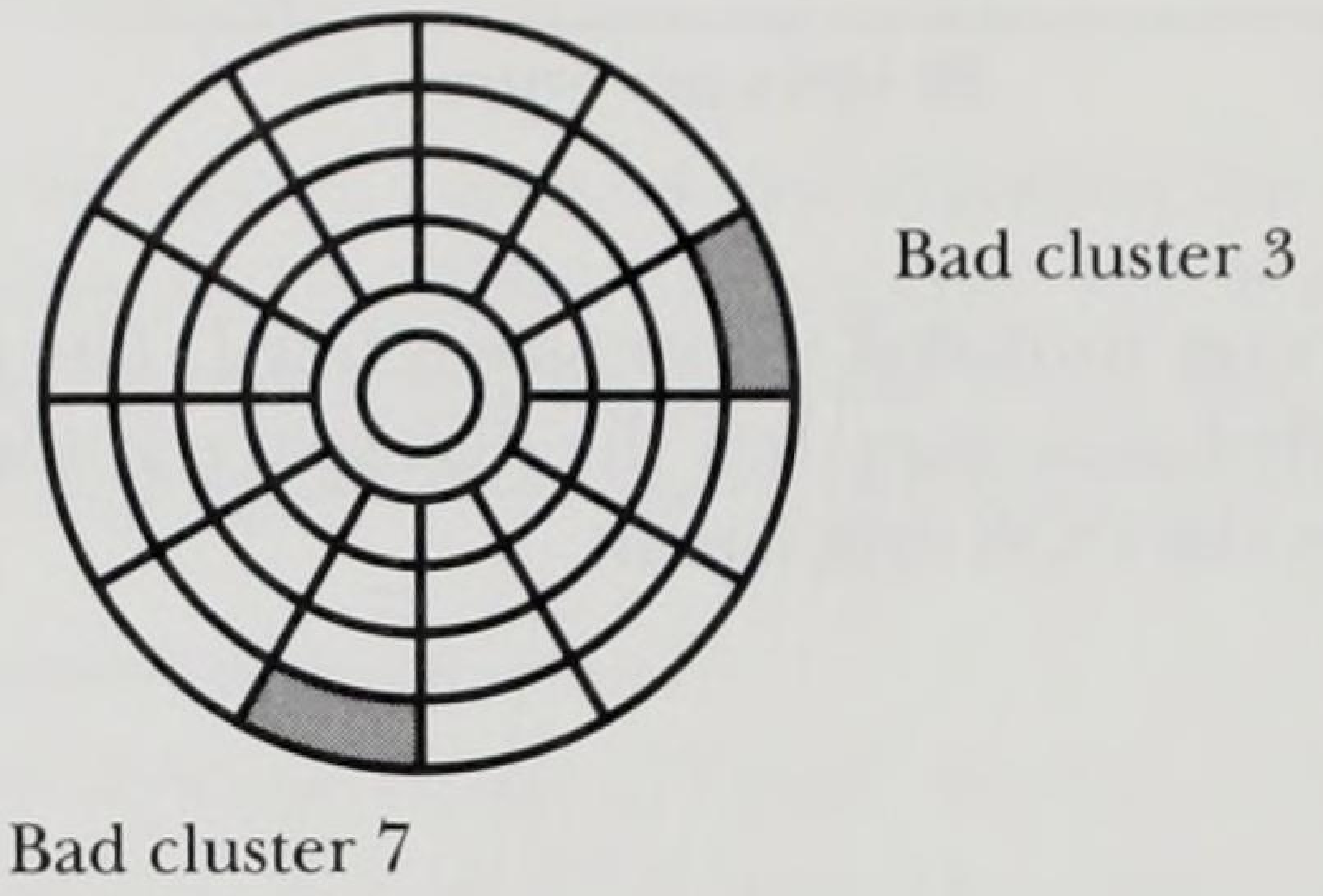

Boot record (1 sector)
Two file allocation tables (number of sectors depends on disk size)
Data space
Root directory (4 to 32 sectors)

Figure 33-5. Each disk contains a boot record, file allocation tables, a root directory, and data space



Cluster	Next cluster
:	:
3	FFF7
:	:
7	FFF7
:	:

File allocation table

Figure 33-6. *FORMAT stores the value FFF7 in the file allocation table to indicate an unusable cluster*

Before FORMAT prepares an unused disk for use, the disk is blank. FORMAT not only searches the disk for bad sectors, FORMAT builds the disk's boot record and the file allocation tables, as well as the root directory.

How CHKDSK Works

You learned in Chapter 25 that CHKDSK examines your disk's current health. CHKDSK performs a consistency check between the information stored about files in directory entries and the information stored in the file allocation table. To begin, CHKDSK reads a directory entry, noting a file's starting cluster and file size. Using the file allocation table entries, CHKDSK first makes sure the file size in the directory corresponds to the number of clusters it found. If the sizes differ, CHKDSK will truncate the file size, displaying the message shown here:

```
Allocation error, size adjusted
```

Next, CHKDSK keeps track of which clusters each file believes it owns (which clusters CHKDSK found in the file allocation chain for the file). If two or more files believe they own one or more of the same clusters, the files are cross-linked as discussed in Chapter 25. When CHKDSK encounters cross-linked files, it displays a message similar to the following:

```
FILENAME.EXT is cross linked on Cluster n
```

Last, CHKDSK searches the file allocation table for clusters marked as in use, but not belonging to a file. As discussed in Chapter 25, CHKDSK calls such clusters *lost clusters.* When CHKDSK encounters lost clusters, CHKDSK displays the following message, telling you the number of lost cluster chains and the amount of disk space the lost cluster chains consume:

```
nnn Lost allocation units found in n chains
nnn bytes disk space would be freed
Convert lost chains to files (Y/N)?
```

As you can see, CHKDSK performs much of its processing simply by comparing the file allocation table entries to the information recorded in directory entries.

SUMMARY

File Allocation Table

The file allocation table, or FAT, keeps a record of each cluster that includes a "next cluster" value. This value is either the number of the next cluster in the chain of the file, FFF8 if the current cluster is the last in the file, 0 if the cluster is empty, or FFF7 if the current cluster is unusable.

The FORMAT command places the FFF7 value in the file allocation table when it encounters a cluster with bad sectors. CHKDSK compares the file allocation table value with the directory entry value to find cluster size errors and cross-linked and lost clusters.

The file allocation is stored at the beginning of the disk, immediately following the boot record. Since the file allocation is so vital, DOS stores two copies of it.

Directory Entries

Each time you create a file, DOS creates a directory entry for it, which includes the file's name, extension, size, date and time stamps, and the number of the file's first cluster.

Review

1. What is the file allocation table (FAT)?
2. Given the following file allocation table and directory entries in Figure 33-7 mark the disk clusters in use by each file.
3. How does DOS find unused clusters?
4. How does FORMAT prevent DOS from using a damaged cluster?
5. A 360Kb disk can store 368,640 bytes of data. However, DOS uses portions of this disk to store the boot record, FAT, and root directory. How much disk space is actually available for files?

Answers

1. The file allocation table provides the road map DOS uses to locate files on your disk. To store a file on disk, DOS divides the file into one or more disk clusters. The file allocation table tracks the list of clusters that make up each

Name	Size	Date	Time	Starting cluster
ONE	2048	8-1-93	6:01p	4
TWO	4096	8-1-93	6:02p	3
THREE	6144	8-1-93	6:03p	6

Cluster	Next cluster
2	0
3	5
4	FFF8
5	FFF8
6	7
7	8
8	FFF8

File allocation table

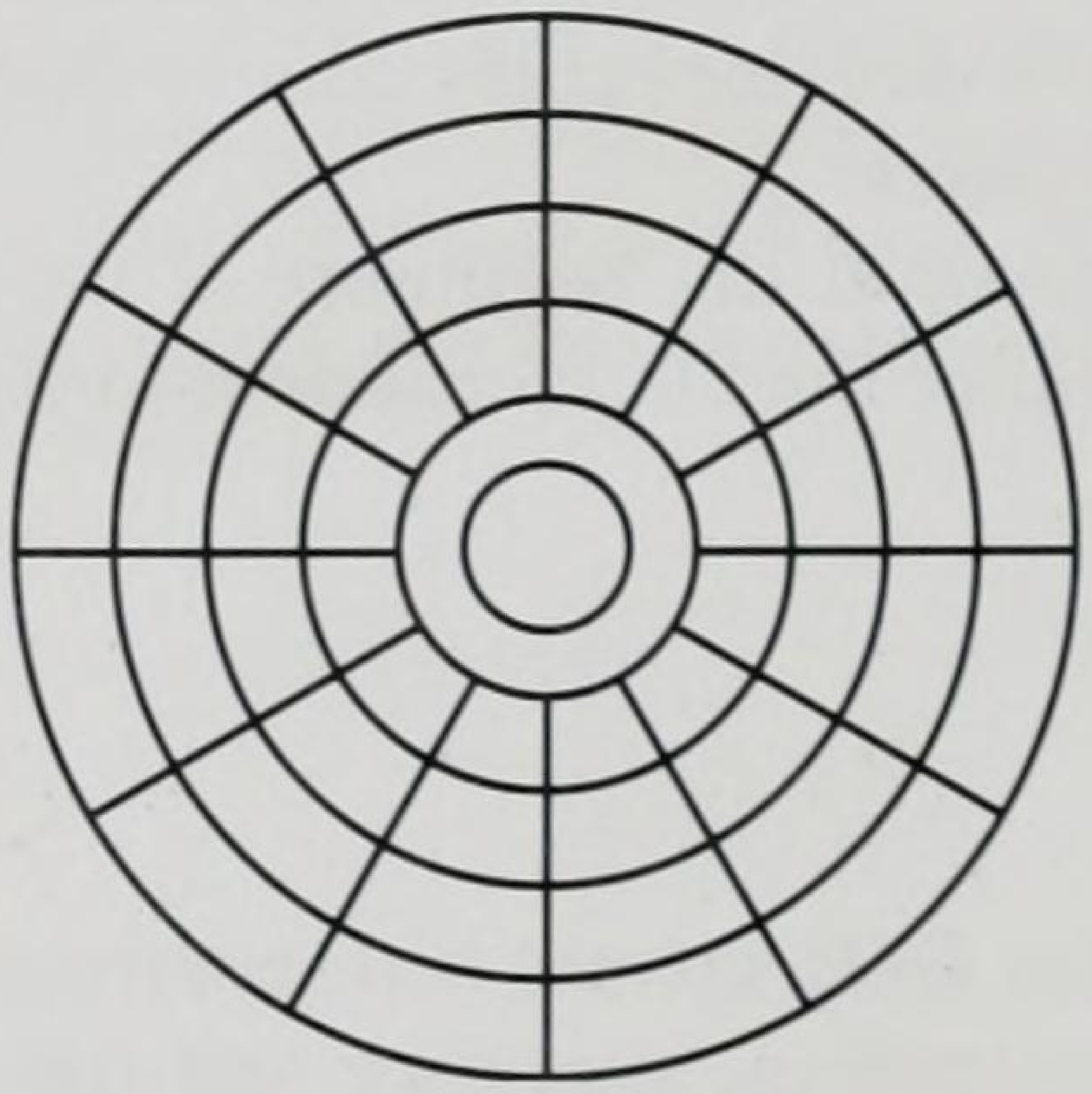


Figure 33-7. *Marking the disk clusters used by each file*

file. DOS stores the file allocation table on disk immediately after the disk's boot record. To safeguard against a single disk error within the FAT's sectors preventing you from accessing your files, DOS stores two copies of the file allocation table on disk. If the first copy becomes damaged, DOS can use the second to access your files.

2. The disk clusters for the files are shown in Figure 33-8
3. DOS assigns the value 0 to unused clusters within the file allocation table. When you create a file or increase an existing file's size, DOS searches the file allocation table for a cluster entry with the value 0.
4. FORMAT places the hexadecimal value FFF7 in the file allocation table for damaged clusters. When DOS searches the file allocation table for an available cluster, DOS knows to avoid clusters marked by FORMAT as unusable.
5. The following screen shows the amount of space available for files using a 360Kb disk:

```
Boot record requires 512 bytes
File allocation tables require 2,048 bytes
Root directory requires 3,584 bytes

File space is left with 362,496 bytes
```

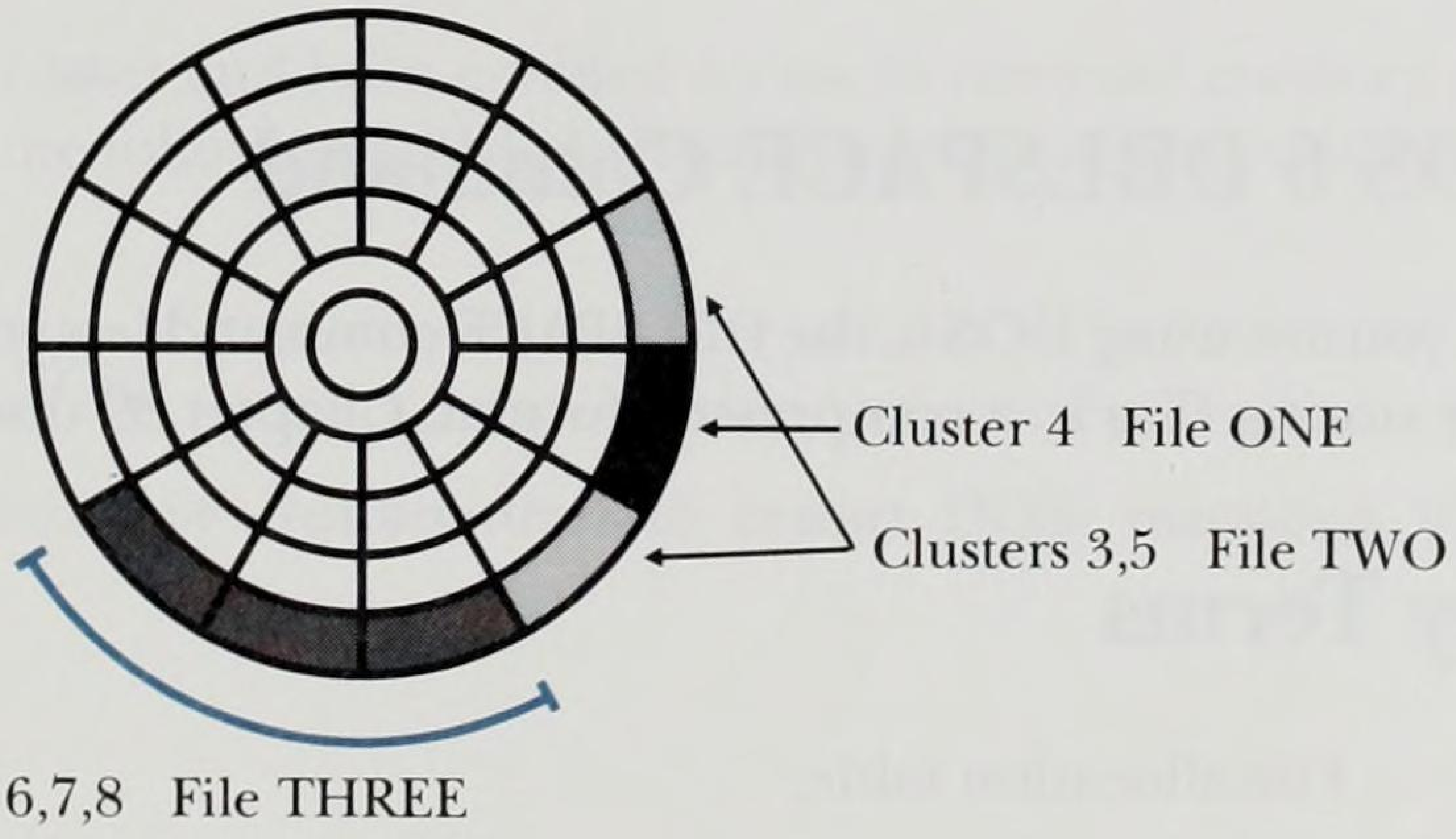


Figure 33-8. *File cluster assignment*

Advanced Concepts

As you have learned, one of the major problems arising from DOS assigning disk space to files using clusters is wasted space for small files. One of the largest offenders of wasted disk space is DOS batch files. If you count up the number of batch files on your disk and multiply the number of files by your disk's cluster size, you can determine how much space your batch files consume. If you are using DOS 3.3 or later, use the following command to determine the number of batch files that are on your disk:

```
C:\> ATTRIB  \*.BAT  /S  |  FIND  /C  ".BAT" <ENTER>
```

If you are using DOS 5 or later, you can quickly determine the amount of space your batch files consume using DIR as follows:

```
C:\> DIR \*.BAT  /S <ENTER>
```

If you have several batch files you don't use on a regular basis, you might consider merging the batch files and using either a menu or batch parameters to select a specific batch file from the resulting file. By merging files in this way, you can quickly free up disk space.

If you are using DOS 5 or later, you might consider defining your commonly used smaller batch files as DOSKEY macros. To do so, you would place DOSKEY commands in your AUTOEXEC.BAT file, directing DOSKEY to define the macros each time your system starts. Using macros, you trade memory for disk space.

DOS 6 DBLSPACE Command

If you are using DOS 6, the DBLSPACE command lets you double your disk's capacity by storing files in a compressed format. Chapter 39 discusses DBLSPACE in detail.

Key Terms

FAT File allocation table.

File allocation table A table containing the locations of each file's clusters, of damaged clusters marked unusable by FORMAT, and of available clusters DOS can use to store additional information on disk.

Chapter 34

ANSI.SYS

In Chapter 24 you learned that the CONFIG.SYS DEVICE entry lets you install device-driver software into memory. At that time you learned that DOS provides several different device drivers, one of which, ANSI.SYS, lets you enhance your screen and keyboard capabilities. Because of its popularity and frequency of use, this chapter takes a detailed look at ANSI.SYS. Specifically, you will learn to use ANSI.SYS to set your screen colors, to redefine function keys, and to improve the output appearance of your DOS batch files.

Installing ANSI.SYS

ANSI.SYS is a device driver that resides in the DOS directory on your hard disk. Using your word processor or the DOS editor, add the following entry to your CONFIG.SYS file:

```
DEVICE=C:\DOS\ANSI.SYS
```

If you are using DOS 5 or later and have enabled access to reserved memory as discussed in Chapter 32, use the following DEVICEHIGH entry to load the ANSI.SYS driver:

```
DEVICEHIGH=C:\DOS\ANSI.SYS
```

Next, use the CTRL-ALT-DEL key combination to restart DOS, installing the ANSI.SYS device driver.

Escape Sequences

The ANSI.SYS driver lets you set screen colors, redefine keys, and position the cursor by writing unique character combinations to your screen. These character combinations begin with the ASCII *escape character* (ASCII value 27) and are called *escape sequences*. The escape character is a unique character that does not have a corresponding symbol that appears on your screen. Most books and magazines represent the escape character as Esc. Using the word "Esc", we can represent the ANSI escape sequence Esc[44m that sets your screen background color to blue. In this case, the escape sequence contains the escape character immediately followed by a left bracket ([) and the characters 44m. This chapter presents many ANSI escape sequences. Each escape sequence will begin with "Esc" and a left bracket ([).

Representing the Escape Character

In general, the ANSI device driver examines every character DOS writes to your screen display. When the ANSI driver encounters an escape character, the driver checks the characters that follow to see if they match one of its defined escape sequences.

To produce an ANSI escape sequence, you can use your word processor or the DOS editor to create a file containing the escape sequence, which you later display using the TYPE command.

```
C:\> TYPE  BLUEBACK.ESC <ENTER>
```

Second, you can create a batch file that uses ECHO to write the escape sequence to your screen.

```
@ECHO OFF
ECHO Esc[44m
```

Your difficulty in using either of these techniques is entering the Esc character. When you are editing a text file or batch file, you can't just press the ESC key to generate the escape character. Instead, depending on your word processor, the steps you must perform will differ. If you simply type **ECHO Esc[44m**, ECHO will dutifully echo "Esc[40m" on screen. The following discussion teaches you how to create the batch file BLUEBACK.BAT, which uses the ECHO command to write the ANSI escape sequence Esc[44m to your screen, setting your screen's background color to blue.

```
@ECHO OFF
ECHO Esc[44m
CLS
```

Creating Escape Sequences with EDLIN

To create the Esc character using EDLIN, you must press CTRL-V (hold down the CTRL key and type **V**). Next, you must type a left bracket character. Issue the following command to create the file BLUEBACK.BAT:

```
C:\> EDLIN  BLUEBACK.BAT <ENTER>
```

EDLIN will respond with the "New file" message and its asterisk (*) prompt.

```
New file
*
```

Type **I** and press ENTER to type in line 1, and then enter the batch file's first line as follows:

```
New file
*I <ENTER>
        1:*@ECHO OFF <ENTER>
        2:*
```

Next, type **ECHO** followed by a space. To generate the Esc character, use the CTRL-V key combination, followed by a left bracket. Your screen will display only the "^V" as shown here:

```
New file
*I <ENTER>
        1:*@ECHO OFF <ENTER>
        2:*ECHO ^V
```

You have now generated only the Esc character, and you must now complete the escape sequence by typing **[44m**

```
New file
*I <ENTER>
        1:*@ECHO OFF <ENTER>
        2:*ECHO ^V[44m
```

Press ENTER and type **CLS** for line 3.

```
New file
*I <ENTER>
        1:*@ECHO OFF <ENTER>
        2:*ECHO ^V[44m <ENTER>
        3:*CLS <ENTER>
        4:*
```

The batch file is complete. Use the CTRL-C key combination to leave EDLIN's insert mode and the EDLIN E command to exit EDLIN to DOS.

```
New file
*I <ENTER>
        1:*@ECHO OFF <ENTER>
        2:*ECHO ^V[44m <ENTER>
        3:*CLS <ENTER>
        4:*^C

*E <ENTER>

C:\>
```

Your batch file is ready for execution.

```
C:\> BLUEBACK <ENTER>
```

If your screen color does not change, make sure you have installed the ANSI.SYS device driver using CONFIG.SYS, and, if necessary, restarted DOS. Next, delete the batch file and use EDLIN to create a second copy, making sure you type in the escape sequence exactly the same as just described.

Creating Escape Sequences with EDIT

Chapter 37 examines the EDIT command in detail. To create an escape sequence using EDIT, press the CTRL-P key combination and then press ESC. EDIT will display a left arrow. Type in the rest of the escape sequence immediately next to the arrow.

Creating Escape Sequences with a Word Processor

If you are using a word processor to create the batch file, the steps you must follow to create the Esc character will differ. In Microsoft Word, make sure your NUMLOCK key is on, and then hold down the ALT key and type **27** *on the numeric keypad at the right of your keyboard.* When you release the ALT key, Word will display the Esc character as a left arrow.

In WordPerfect, like Microsoft Word, hold down ALT and type **27** on your keyboard's numeric keypad. WordPerfect will display the Esc character as a left-pointing arrow. If you are using a different word processor, refer to the documentation that accompanied the software. No matter which word processor you use, make sure you save the file as an ASCII file, not a word processing document file with special embedded formatting characters and symbols.

Controlling Screen Colors Using ANSI.SYS

When you use ANSI.SYS to set your screen color, the color will remain in effect while you issue commands such as DIR, TYPE, and CHKDSK from the DOS prompt. If you invoke a program such as your word processor or the DOS 5 or later EDIT command, the program will override your current screen color selection, using its own colors. If you don't like the program's colors, most programs provide an option that lets you select a color you like. When you exit the program to DOS, your ANSI color selection will resume.

The ANSI.SYS device driver provides foreground and background color values. As you just learned, the escape sequence Esc[44m lets you set your screen's background color to blue. By replacing the color value 44 with one of the values specified in Table 34-1, you can select a different background color.

The following batch file, REDBACK.BAT, selects a red screen background:

```
@ECHO OFF
ECHO Esc[41m
CLS
```

Value	Background Color	Escape Sequence
40	Black	Esc[40m
41	Red	Esc[41m
42	Green	Esc[42m
43	Yellow	Esc[43m
44	Blue	Esc[44m
45	Magenta	Esc[45m
46	Cyan	Esc[46m
47	White	Esc[47m

Table 34-1. *ANSI.SYS Background Color Values*

Value	Background Color	Escape Sequence
30	Black	Esc[30m
31	Red	Esc[31m
32	Green	Esc[32m
33	Yellow	Esc[33m
34	Blue	Esc[34m
35	Magenta	Esc[35m
36	Cyan	Esc[36m
37	White	Esc[37m

Table 34-2. *ANSI.SYS Foreground Color Values*

ANSI.SYS also lets you set your screen's foreground color. Table 34-2 contains the ANSI.SYS foreground color values.

For example, the following batch file, GREENFOR.BAT, selects green for a foreground color:

```
@ECHO OFF
ECHO Esc[32m
CLS
```

Most users will select a foreground and background color combination they find most pleasing to work with. By separating the two color values with a semicolon, the ANSI.SYS device driver lets you select a foreground and background color in one escape sequence. The following batch file, WHITEBLU.BAT, selects a white background (47) and a blue foreground (34):

```
@ECHO OFF
ECHO Esc[47;34m
CLS
```

Controlling Screen Colors with PROMPT

There is one more way to control the screen colors with the Esc characters that you can use either in a predefined batch file (including AUTOEXEC.BAT) or interactively from the DOS prompt, namely through use of the PROMPT command. In Chapter 10 you learned the letters PG direct PROMPT to use the current drive letter and directory name, followed by a greater-than symbol for your system prompt. In a similar way, $E directs PROMPT to generate the Esc character. The following PROMPT command chooses a black background and a magenta foreground:

```
C:\>PROMPT $E[40;35m$P$G <ENTER>
```

You can see that the format of the color command from the [to the m is the same as before. Using the $E character, the next PROMPT uses the escape sequence Esc[44m to display your prompt on a blue background, and the escape sequence Esc[40m to display your other screen text on black:

```
C:\>PROMPT $E[44m$P$G$E[40m <ENTER>
```

PROMPT is very useful for this purpose because you can type it right in from the DOS prompt and see the results immediately. It is limited in its use of escape characters to the colors, however. When you find a combination you like, you can include it in AUTOEXEC.BAT and other batch files. Be careful when typing. If you choose the same number twice, the foreground and background will be the same color—the characters you type will be invisible, and you may have to reboot DOS.

By combining ANSI escape sequences with the prompt *metacharacters* ($ character combinations) presented in Chapter 43, you can create very unique custom prompts.

Redefining Keys with ANSI.SYS

The ANSI.SYS device driver enhances your screen and keyboard capabilities. As you have just learned, using ANSI.SYS you can select your screen's foreground and background colors. In this section, you will use ANSI.SYS to assign your commonly used commands to function keys and different key combinations.

Assume, for example, you use the following TREE command on a regular basis:

```
C:\> TREE  C:\  /F <ENTER>
```

Rather than constantly retyping the command, you can use ANSI.SYS to assign the command to the F10 function key. Each time you press F10 at the DOS prompt, the ANSI.SYS driver will display the TREE command, letting you press ENTER to execute the command.

To assign a command to a key using ANSI.SYS, you must know the key's unique *scancode value* that the keyboard sends to DOS each time the key is pressed. For example, when you press the F10 function key, the keyboard sends DOS the value 0 followed by the value 68. Using these two values, the batch file, F10TREE.BAT, shown in the following example, assigns the TREE command to the F10 function key.

```
@ECHO OFF
ECHO Esc[0;68;"TREE C:\ /F"p
```

In general, you should assign commands to key combinations such as SHIFT-F10 or CTRL-F10 to avoid conflicts with other commands such as DOSKEY. Table 34-3 lists the scancode values for each key on your keyboard, as well as the scan codes generated when you use the key in combination with SHIFT, CTRL, or ALT.

Key	Key only	SHIFT+Key	CTRL+Key	ALT+Key
F1	0;59	0;84	0;94	0;104
F2	0;60	0;85	0;95	0;105
F3	0;61	0;86	0;96	0;106
F4	0;62	0;87	0;97	0;107
F5	0;63	0;88	0;98	0;108
F6	0;64	0;89	0;99	0;109
F7	0;65	0;90	0;100	0;110
F8	0;66	0;91	0;101	0;111
F9	0;67	0;92	0;102	0;112
F10	0;68	0;93	0;103	0;113
F11	0;133	0;135	0;137	0;139
F12	0;134	0;136	0;138	0;140
HOME	0;71	55	0;119	
UP ARROW	0;72	56	0;141	
PGUP	0;73	57	0;132	
LEFT ARROW	0;75	52	0;115	
RIGHT ARROW	0;77	54	0;116	
END	0;79	49	0;117	
DOWN ARROW	0;80	50	0;145	
PGDN	0;81	51	0;118	
INS	0;82	48	0;146	
DEL	0;83	46	0;147	
HOME*	224;71	224;71	224;119	224;151
UP ARROW*	224;72	224;72	224;141	224;152
PGUP*	224;73	224;73	224;132	224;153
LEFT ARROW*	224;75	224;74	224;115	224;155
RIGHT ARROW*	224;77	224;77	224;116	224;157

* Indicates a gray key

Table 34-3. *Keyboard Scancode Values*

Key	Key only	SHIFT+Key	CTRL+Key	ALT+Key
END*	224;79	224;79	224;117	224;159
DOWN ARROW*	224;80		224;145	224;160
PGDN*	224;81	224;81	224;118	224;161
INS*	224;82	224;82	224;146	224;162
DEL	224;83	224;83	224;147	224;163
PRTSC			0;114	
PAUSE/BREAK			0;0	
BACKSPACE	8	8	127	0;14
ENTER	13	13	10	0;28
TAB	9	0;15	0;148	0;165
A	97	65	1	0;30
B	98	66	2	0;48
C	99	67	3	0;46
D	100	68	4	0;32
E	101	69	5	0;18
F	102	70	6	0;33
G	103	71	7	0;34
H	104	72	8	0;35
I	105	73	9	0;23
J	106	74	10	0;36
K	107	75	11	0;37
L	108	76	12	0;38
M	109	77	13	0;50
N	110	78	14	0;49
O	111	79	15	0;24
P	112	80	16	0;25
Q	113	81	17	0;16
R	114	82	18	0;19
S	115	83	19	0;31
T	116	84	20	0;20
U	117	85	21	0;22
V	118	86	22	0;47
W	119	87	23	0;17

* Indicates a gray key

Table 34-3. Keyboard Scancode Values (continued)

Key	Key only	SHIFT+Key	CTRL+Key	ALT+Key
X	120	88	24	0;45
Y	121	89	25	0;21
Z	122	90	26	0;44
1	49	33		0;120
2	50	64		0;121
3	51	35		0;122
4	52	36		0;123
5	53	37		0;124
6	54	94	30	0;125
7	55	38		0;126
8	56	42		0;127
9	57	40		0;128
0	48	41		0;129
-	45	95	31	0;130
=	61	43		0;131
[	91	123	27	0;26
\	92	124	28	0;43
]	93	125	29	0;27
;	59	58		0;39
’	39	34		0;40
,	44	60		0;51
.	46	62		0;52
/	47	63		0;53
Keypad ENTER	13	13	10	0;166
Keypad /	47	47	0;149	0;164
Keypad *	42	42	0;150	0;55
Keypad -	45	45	0;142	0;74
Keypad +	43	43	0;144	0;78
Keypad 5	0;76	53	0;143	

* Indicates a gray key

Table 34-3. *Keyboard Scancode Values* (continued)

Using Table 34-3, the following batch file assigns the command DIR | MORE to the ALT-F10 key combination:

```
@ECHO OFF
ECHO Esc[0;113;"DIR | MORE"p
```

Figure 34-1 illustrates a 101-key keyboard. If you have purchased an enhanced 101-key keyboard after buying your computer, the computer may not fully support some of the extended keys. If you experience difficulties when you are using the scan codes for the extended keys, and are using DOS 4 or later, you can include the /K switch in the CONFIG.SYS DEVICE entry for ANSI.SYS to direct ANSI.SYS to ignore the extended keys.

```
DEVICE=C:\DOS\ANSI.SYS  /K
```

The 101-key keyboard contains several duplicate keys, such as two HOME, END, PGUP, and PGDN keys. By default, DOS treats these keys as the same. If you want ANSI.SYS to assign one of these keys a character string as just discussed and you are using DOS 4 or later, place the /X switch in the CONFIG.SYS DEVICE entry for ANSI.SYS, as shown here:

```
DEVICE=C:\DOS\ANSI.SYS  /X
```

Creating ANSI.COM to Execute Escape Sequences

The previous examples have used the ECHO command within DOS batch files to write escape sequences. Once you learn how to generate the Esc character using your word processor or the DOS editor, creating such batch files is fairly easy.

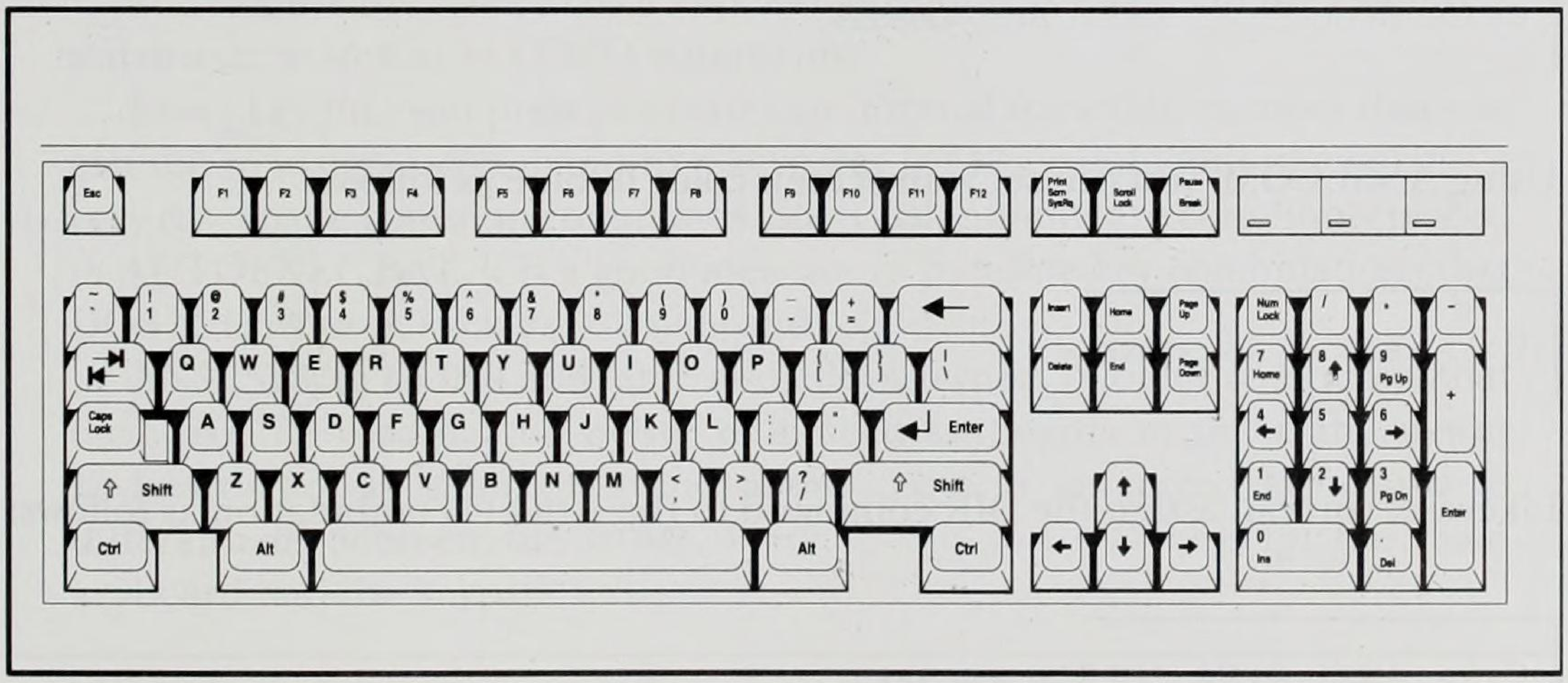


Figure 34-1. A 101-key keyboard

In Chapter 29 you learned how to use DEBUG script files to create custom commands. The following script file, ANSI.SCR, creates the command ANSI.COM that makes it easier for you to write escape sequences:

```
N ANSI.COM
A 100
MOV AH, 2
MOV SI, 82
MOV DL, 1B
INT 21
MOV DL, 5B
INT 21
MOV DL, [SI]
CMP DL, D
JZ 119
INT 21
INC SI
JMP 10D
MOV AH, 4C
INT 21
<ENTER>
R CX
1D
W
Q
```

Create the file ANSI.COM using the DOS input redirection operator, as shown here:

```
C:\> DEBUG  <  ANSI.SCR <ENTER>
```

Using ANSI.COM, you can set your screen color to blue as follows:

```
C:\> ANSI 44m <ENTER>
```

Likewise, you can assign the DIR command to the SHIFT-F1 combination as follows:

```
C:\> ANSI 0;84;"DIR"p <ENTER>
```

So far, each time you have redefined a key to a commonly used function, pressing the key has only displayed the command name, forcing you to press ENTER to execute the command. If you include the carriage return character (ASCII 13) in the key's definition, DOS will automatically execute the command each time you press the key. The following command includes the carriage return character in the key definition, directing DOS to execute the DIR command each time you press SHIFT-F1:

```
C:\> ANSI 0;84;"DIR";13p <ENTER>
```

SUMMARY

ANSI.SYS

ANSI.SYS is a device driver that resides in the DOS directory of your hard disk. It allows you to enhance your screen and keyboard capabilities. Before you can use ANSI.SYS you must load it into memory with a CONFIG.SYS DEVICE= (or, if you have enabled access to reserved memory in DOS 5 or later, DEVICEHIGH=) statement.

You can set foreground and background colors, set other text attributes (such as blinking or bold), redefine keypresses so that a single keypress (or combination) can trigger a command, and position the cursor anywhere on the screen by using escape sequences, statements that use the Esc character. Since the Esc character can't be pressed directly and typing **Esc** only produces the word "Esc" (not the character), you must use a word processor or text editor to produce the Esc character.

You can use the screen color and text attribute characters in the PROMPT statement, as well as in ECHO statements.

Every key that you press generates an internal scancode number that you can use to assign a command to a keypress (or key combination), making it very easy to use common commands. Such assignments are commonly placed in AUTOEXEC.BAT. It is a good idea not to redefine key combinations that DOS already uses, such as SHIFT-F10 and CTRL-F10.

ANSI.SYS, in DOS 4 and later, also allows owners of older computers that have trouble reading the newer enhanced keyboards to make the newer keyboards act as older ones and allows users of enhanced keyboards to differentiate between the HOME, PGUP, PGDN, and END keys on the main keyboard and the keypad.

Hands On

Using your word processor, place a DEVICE or DEVICEHIGH entry in your CONFIG.SYS file and reboot DOS to load the ANSI driver, if you have not already done so.

Next, create the batch file YELLBLUE.BAT that selects a yellow screen background and a blue foreground.

```
@ECHO OFF
ECHO Esc[43;34m
CLS
```

If you don't know how to create the Esc character with your word processor, build the command ANSI.COM using the file ANSI.SCR, as previously discussed. Change the batch file to use ANSI.COM as follows:

```
@ECHO OFF
ANSI 43;34m
CLS
```

Using your ANSI.COM command, assign CHKDSK to the ALT-C key combination.

```
C:\> ANSI  0;46;"CHKDSK"p <ENTER>
```

Using the ALT-C key combination and ENTER, invoke CHKDSK.

Next, change the previous escape sequence slightly using the value 13 (ENTER key) to direct ANSI.SYS to automatically execute CHKDSK when you press ALT-C without waiting for you to press ENTER

```
C:\> ANSI  0;46;"CHKDSK";13p <ENTER>
```

Review

1. When should you use DEVICEHIGH instead of DEVICE to install ANSI.SYS?
2. What is an escape sequence?
3. How do you generate the Esc character?
4. What is the function of ANSI.COM?
5. Use ANSI.COM to assign the CLS command to F10. Use the value 13 to direct ANSI.SYS to automatically execute the command without forcing you to press the ENTER key.

Answers

1. If you are using DOS version 5 or later, and your CONFIG.SYS file contains the DOS=UMB entry and loads the EMM386.EXE device driver as discussed in Chapter 32, you can save conventional memory by directing DOS to load the ANSI.SYS device driver into reserved memory using DEVICEHIGH.
2. An escape sequence is a series of two or more characters, the first of which is the ASCII escape character (ASCII value 27). The ANSI.SYS device driver lets you use different escape sequences to set your screen attributes and to redefine keyboard keys.
3. If you are using EDLIN, press CTRL-V and then type [(a left bracket). If you are using EDIT, press CTRL-P and then press ESC. If you are using a word processor, refer to the documentation that accompanied your software to determine the steps you must perform to create an escape character.
4. ANSI.COM simplifies the process of writing ANSI escape sequences. Specifically, ANSI.COM writes the Esc character and left bracket. If you want to use ANSI.SYS to set your screen background color to blue, you need to generate the escape sequence Esc[44m. Using ANSI.COM, you can issue the following command:

```
C:\> ANSI 44m
```

5. The command appears as follows:

```
C:\> ANSI  0;68;"CLS";13p <ENTER>
```

Advanced Concepts

In the previous section you used ANSI.SYS to select screen colors and to define function keys. ANSI.SYS also lets you position the cursor to place output at specific screen locations, select a video display mode, and assign different text attributes.

Cursor Positioning with ANSI.SYS

In Chapter 29 you created a batch file, similar to the one shown in the following example, that displayed a menu at the upper-left corner of your screen.

```
          M E N U

D  List directory files
C  Run CHKDSK
S  Sort directory listing
X  Exit to DOS
Enter choice:
```

To improve the menu's appearance you can change the batch file to display the menu in the middle of your screen. The "brute force" method of placing the menu at the middle of the screen is to use ECHO several times to write blank lines and then to place spaces before each menu option to shift the options to the right.

A better solution, however, is to use ANSI.SYS to place the cursor at specific row and column positions on your screen, before the batch file writes each menu option. For example, the following ECHO command places the cursor at row 10, column 20 and then displays the message:

```
ECHO  Esc[10;20H  Row ten, column twenty
```

By default, screens use 25 rows of 80 columns. Using the row and column position escape sequence, the following batch file changes MENU.BAT slightly to display the menu at the center of your screen:

```
@ECHO OFF
:LOOP
CLS
ECHO Esc[10;30H          M E N U
ECHO.
ECHO Esc[12;30H D  List directory files
ECHO Esc[13;30H C  Run CHKDSK
ECHO Esc[14;30H S  Sort directory listing
ECHO Esc[15;30H X  Exit to DOS
ANSI 17;30H
CHOICE /C:DCSX Enter choice:
IF NOT ERRORLEVEL 1 GOTO EXIT
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO DIR
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO CHKDSK
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO SORT
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO EXIT
:DIR
DIR
PAUSE
GOTO LOOP
:CHKDSK
```

```
CHKDSK
PAUSE
GOTO LOOP
:SORT
DIR | SORT | FIND /V "e" | MORE
PAUSE
GOTO LOOP
:EXIT
ECHO <Alt+255>
```

Table 34-4 describes the ANSI.SYS escape sequence for cursor positioning.

Selecting a Video Mode with ANSI.SYS

Although its use is uncommon, the ANSI.SYS device driver lets you select a video display mode. Table 34-5 lists the available display modes and their corresponding escape sequences.

Escape Sequence	Description	Example
Esc[*Row;Column*H	Places the cursor at the row; column position	Esc[10;20H
Esc[*Rows*A	Moves the cursor up the number of rows specified	Esc[5A
Esc[*Rows*B	Moves the cursor down the number of rows specified	Esc[4B
Esc[*Columns*C	Moves the cursor right the number of columns specified	Esc[20C
Esc[*Columns*D	Moves the cursor left the number of columns specified	Esc[15D
Esc[s	Saves the current cursor row and column position	Esc[s
Esc[u	Restores the cursor to a previously saved row and column position	Esc[u
Esc[2J	Clears the screen and moves the cursor to row 0	Esc[2J
Esc[K	Erases the characters from the current cursor position to the end of the line, leaving the cursor position unchanged	Esc[K

Table 34-4. ANSI.SYS Cursor-Positioning Escape Sequences

Video Mode	Description	Escape Sequence
0	25×40 monochrome text	Esc[=0h
1	25×40 color text	Esc[=1h
2	25×80 monochrome text	Esc[=2h
3	25×80 color text	Esc[=3h
4	320×200 4-color graphics	Esc[=4h
5	320×200 monochrome graphics	Esc[=5h
6	640×200 monochrome graphics	Esc[=6h
7	Line wrap enabled	Esc[=7h
7	Line wrap disabled	Esc[=7l
13	320×200 color graphics	Esc[=13h
14	640×200 16-color graphics	Esc[=14h
15	640×350 monochrome graphics	Esc[=15h
16	640×350 monochrome graphics	Esc[=16h
17	640×480 VGA 16-color graphics	Esc[=17h
18	640×480 VGA 16-color graphics	Esc[=18h
19	320×200 VGA 256-color graphics	Esc[=19h

Table 34-5. *ANSI.SYS Video Mode Escape Sequences*

Assigning Text Attributes with ANSI.SYS

Earlier you learned how to use ANSI.SYS to set your screen's foreground and background colors. Using the red-background escape sequence, the following ECHO command displays a warning message to the user in red and one in black.

```
ECHO  Esc[31mWARNING!Esc[30mAbout to delete files.
```

In addition to letting you display text using different foreground and background colors, the ANSI.SYS device driver lets you use one or more of the text attributes listed in Table 34-6.

The following ECHO command uses blinking text for the word "WARNING!":

```
ECHO Esc[5mWARNING!Esc[0mAbout to delete files.
```

Setting Your Screen Size in DOS 5 or later

If you are using DOS 5 or later and an EGA or VGA screen display, you can use the ANSI.SYS device driver and the MODE command, as discussed in Chapter 43, to

Attribute Value	Description	Escape Sequence
0	Attributes off	Esc[0m
1	Bold	Esc[1m
4	Monochrome underline	Esc[4m
5	Blink	Esc[5m
7	Reverse video	Esc[7m
8	Concealed black text on black background	Esc[8m

Table 34-6. *ANSI.SYS Text-Attribute Values*

specify the number of rows and columns on your screen. Not all of the software programs that you run will support these screen sizes and thus may bypass the ANSI.SYS device driver to use their own screen format.

Key Term

Escape sequences A series of two or more characters, the first of which is the ASCII escape character (ASCII value 27).

Chapter 35

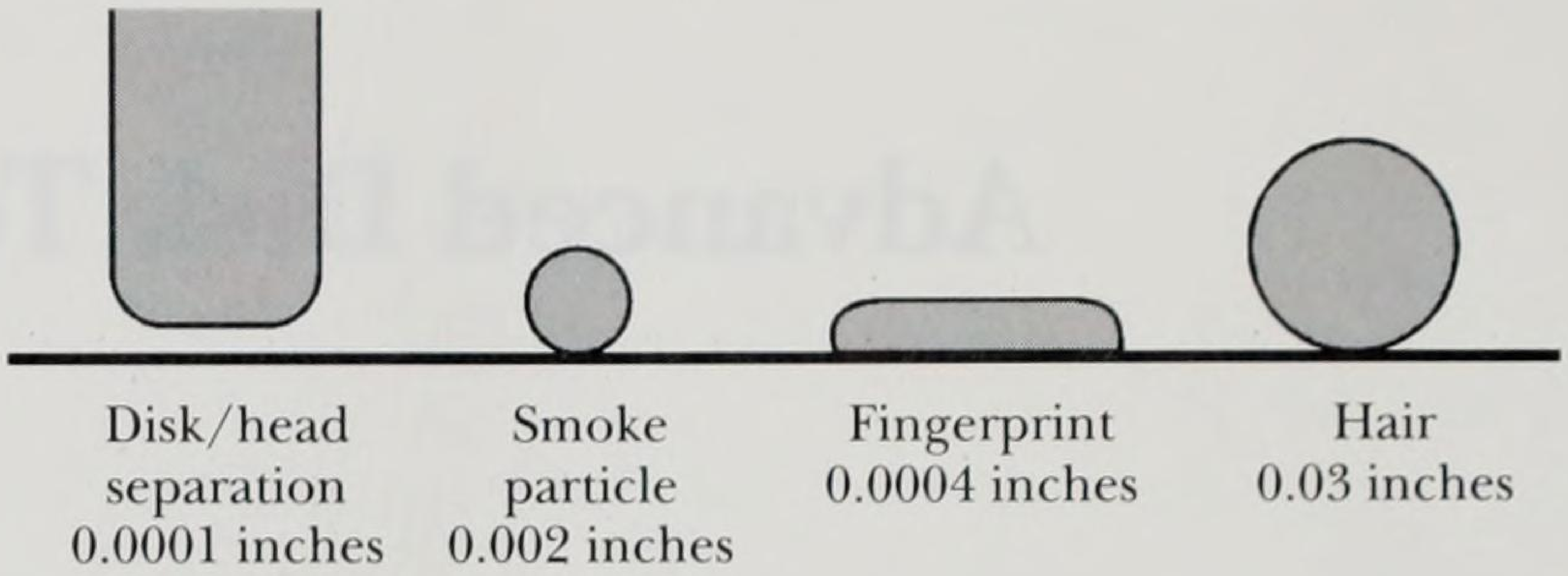
Advanced Disk Topics

Because file and disk management are two of the most critical functions DOS performs, this chapter looks at three important disk concepts: disk head parking, disk sector interleaving, and low-level formatting. By understanding these concepts, you can reduce your chance of damaging your disk, improve your disk performance, and appreciate corrective methods used when your disk experiences errors FORMAT can't recover.

Understanding Disk Head Parking

Your disk rotates past the disk drive's read/write head, as illustrated in Figure 35-1, providing access to every sector on the current track. When the disk drive needs to read or write information to a different track, the drive moves the read/write head in or out as required.

For floppy disks, the read/write head actually makes contact with the disk's surface. For hard disks, however, the read/write heads do not contact the disk's surface, but rather, float just above it. The following illustration shows the distance separating a hard disk surface and the read/write head using several very small objects as a comparison.


Disk/head separation 0.0001 inches
Smoke particle 0.002 inches
Fingerprint 0.0004 inches
Hair 0.03 inches

The hard disk spins past the read/write head at 3600 revolutions per minute (RPMs). If the read/write head accidentally makes contact with the disk's surface, possibly due to the computer being bumped or jarred, a *head crash* occurs. A head crash is generally accompanied by a loud scraping noise as the head scrapes off the disk's specially coated surface, splattering the fragments across the rest of your disk. When a disk head crash occurs, the information the disk contained will very likely be lost and you will probably have to purchase a new disk.

NOTE *There are companies that can disassemble a disk that has experienced a crash, and possibly reconstruct portions of the disk's surviving data. This process is time consuming and very expensive. The best way to avoid loss of data is by performing regular disk backup operations, as discussed in Chapter 22.*

The first way to avoid disk head crashes is to never move your computer while the computer is on and the disk is spinning. When you power off your computer and the

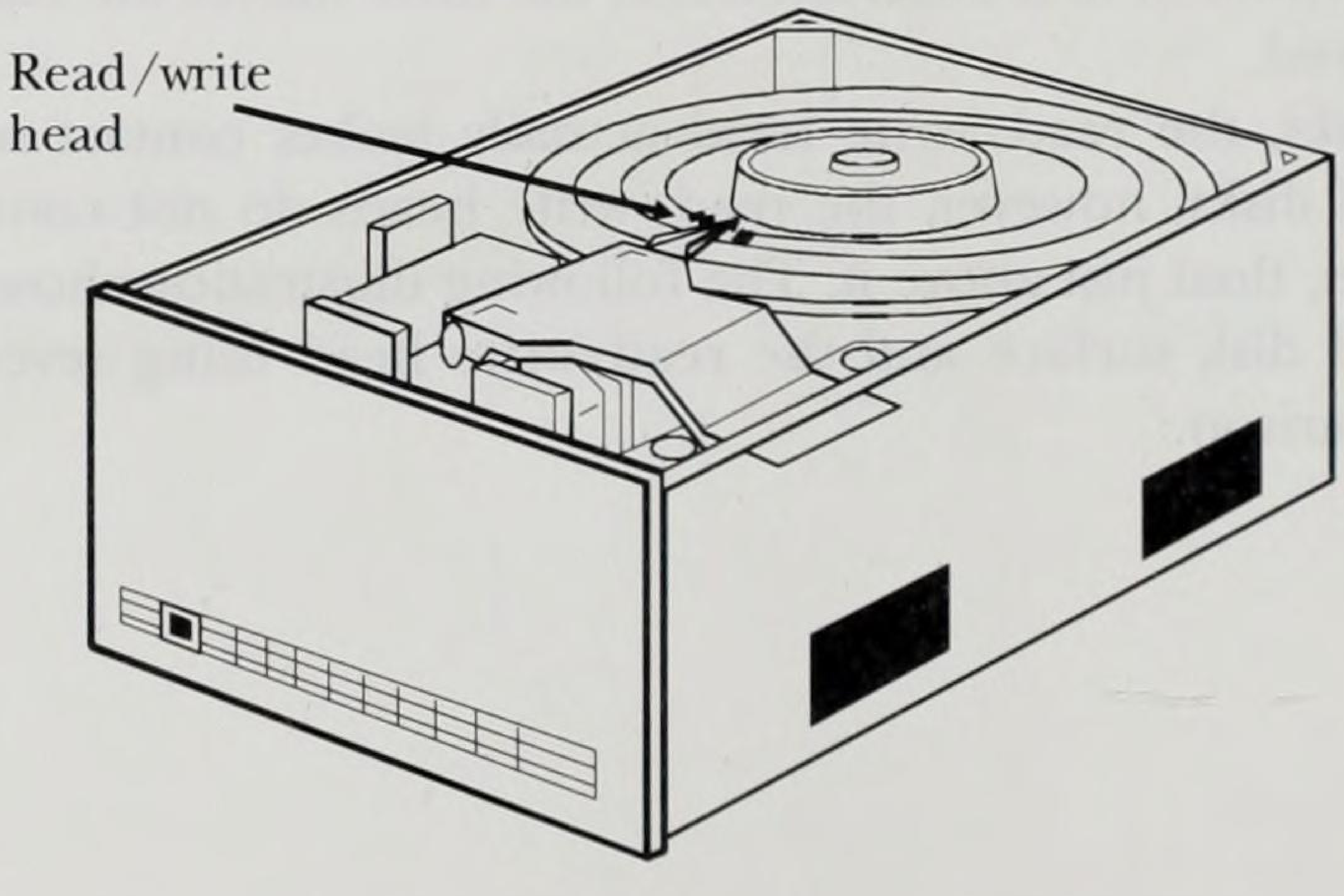


Figure 35-1. *The read/write head moves in and out to access information on the rotating disk*

disk quits spinning, the disk head actually rests on the disk's surface. Because the disk is not moving, a head crash does not occur. When you turn your computer on, the disk drive raises the head off the disk's surface before the disk starts to rotate.

To provide better protection for your disk, many disk drives have a designated *landing zone* as shown in Figure 35-2. When you turn off your computer, the disk controller moves the read/write head above the landing zone before setting the head down on the disk's surface.

For earlier model disk drives that don't automatically move the read/write head to a landing zone, many DOS users use a *park utility* before turning off their computer. Most park commands move the read/write head to the disk's innermost track. If you don't have a disk park utility program, you can create a simple one using DEBUG, which was discussed in Chapter 29.

If you are using an IBM PS/2 computer, create the command PS2PARKC.SCR, shown here. This utility parks only the hard disk drive C.

```
N PS2PARKC.COM
A 100
MOV AH,19
MOV DL,80
INT 13
MOV AH,4C
INT 21
<ENTER>
R CX
A
W
Q
```

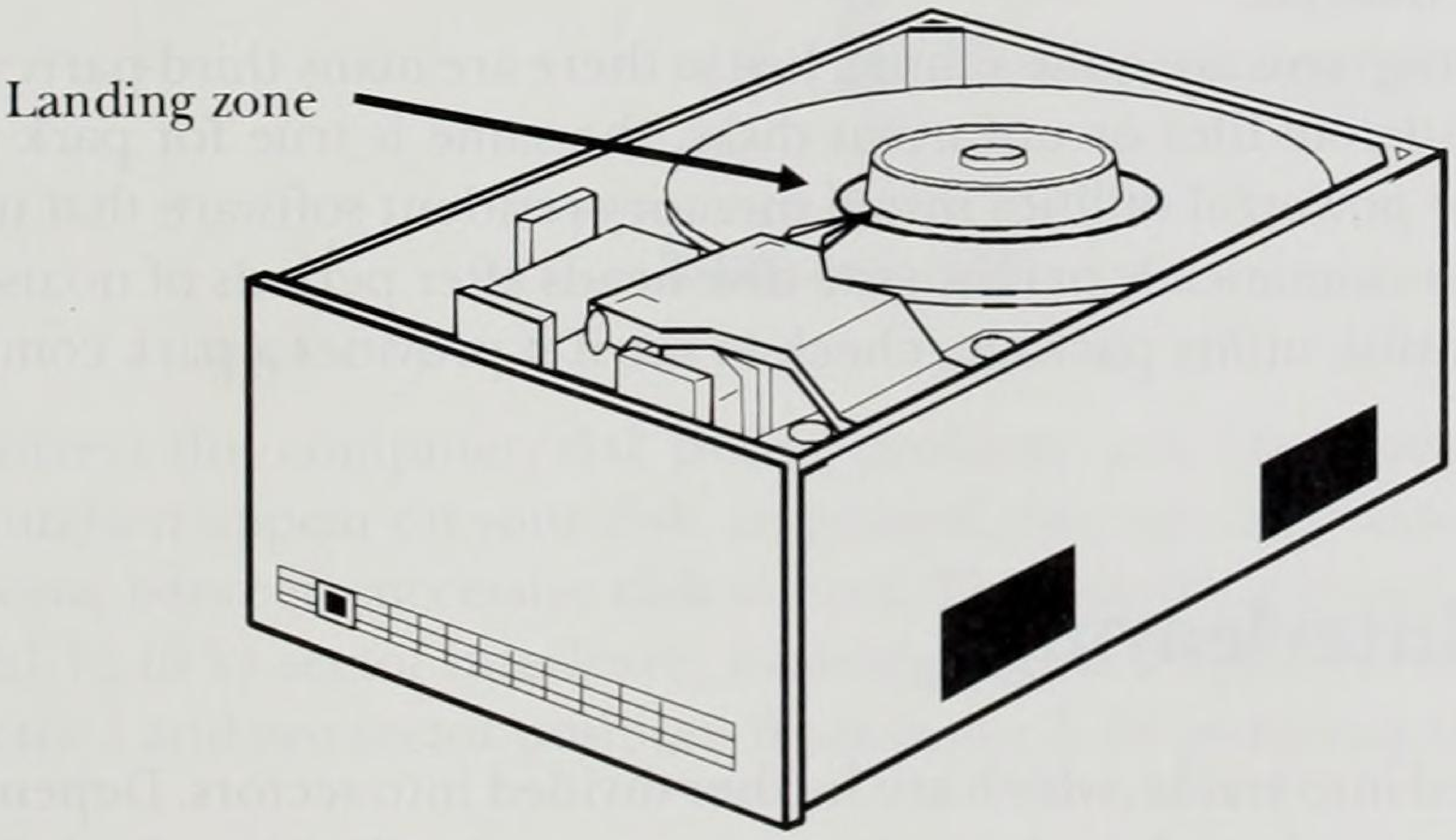


Figure 35-2. *Many disk drives provide a designated landing zone, where the read/write head rests on the disk when the computer is powered off*

If you are using a computer *other* than a PS/2, create the utility PARKC.COM using the DEBUG script file PARKC.SCR, as shown here. This utility also only parks drive C.

```
N PARKC.COM
A 100
MOV AH,8
MOV DL,80
INT 13
JC 110
MOV AH,C
MOV DL,80
INT 13
JMP 117
MOV AH,9
MOV DX,11B
INT 21
MOV AH,4C
INT 21
DB 'Controller failed to respond correctly—heads not parked.$'
<ENTER>
R CX
55
W
Q
```

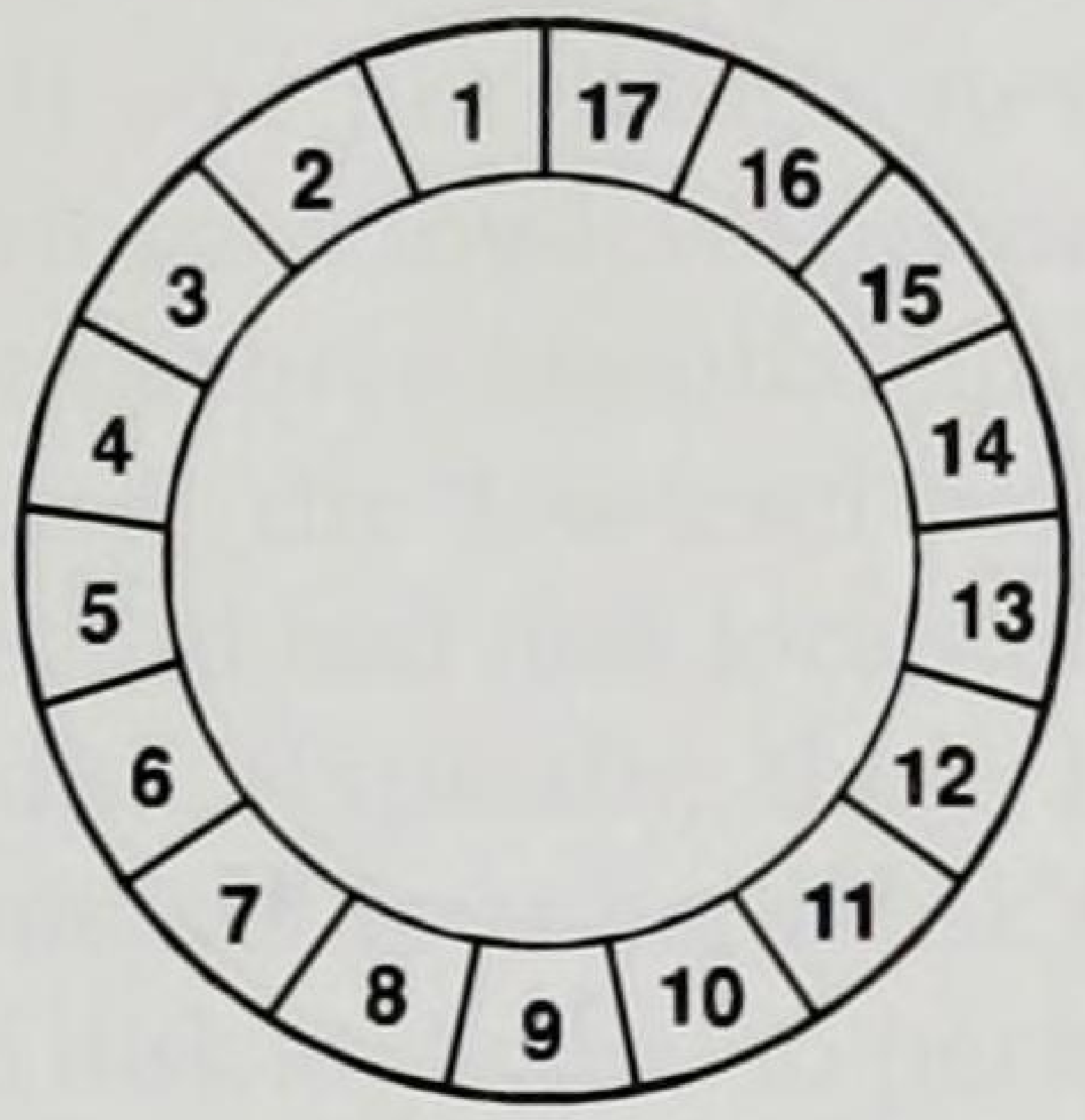
Before you turn off your computer, you can invoke one of these utility programs to move your disk's read/write head over a track that is unlikely to contain data.

Parking your disk head is simply the process of moving the head over a specific track. You can park your disk head any time the disk is not in use. The next time your computer uses the disk, the read/write head automatically moves from the parked track to the track desired.

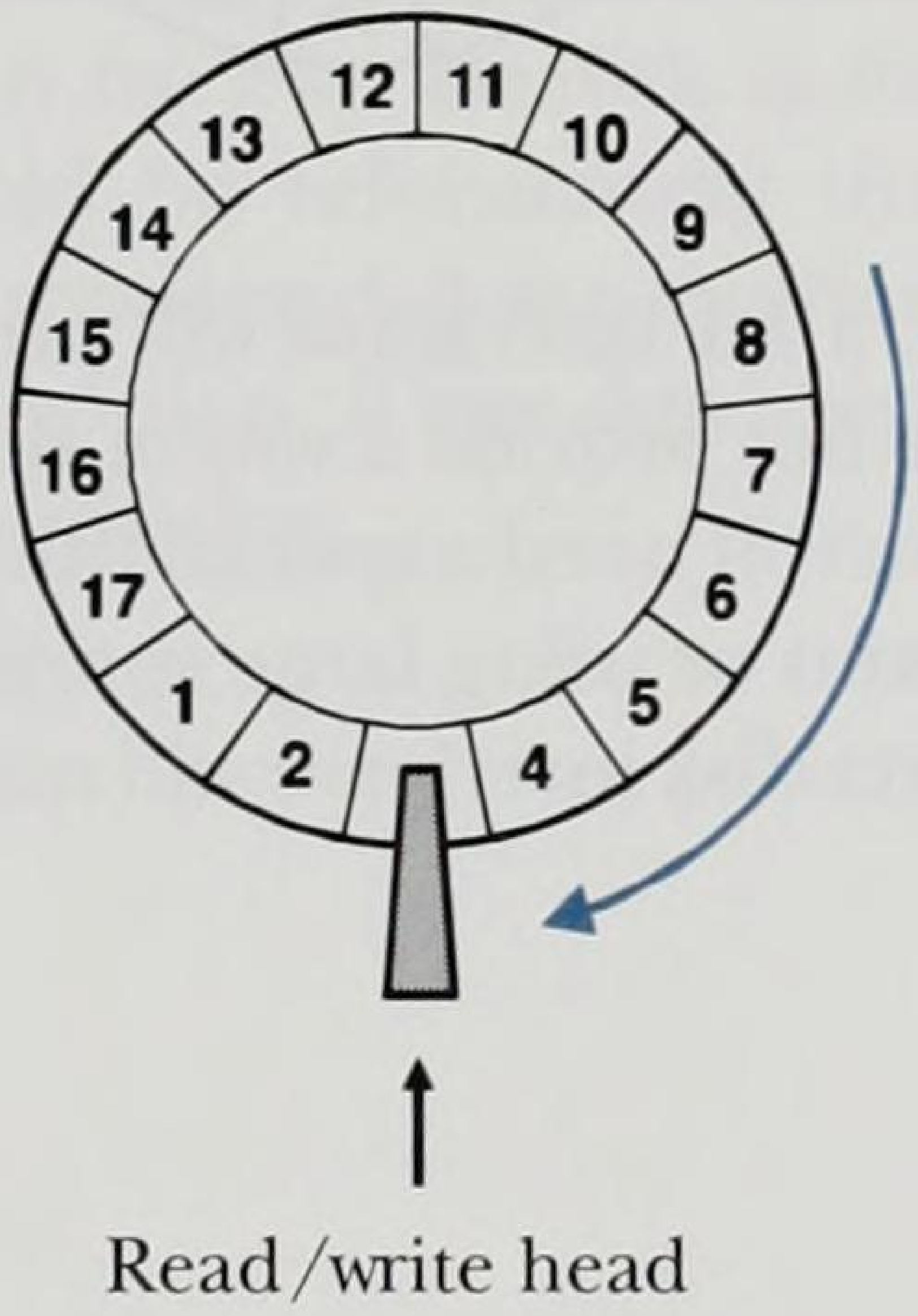
These utility programs are quite simple. Just as there are many third-party software programs that undelete files or unformat disks, the same is true for park utilities. Some of the more powerful utilities install memory-resident software that monitors your disk activity, automatically parking the disk heads after periods of no use. If you have a third-party disk utility package, check to see if it provides a park command.

Disk Sector Interleaving

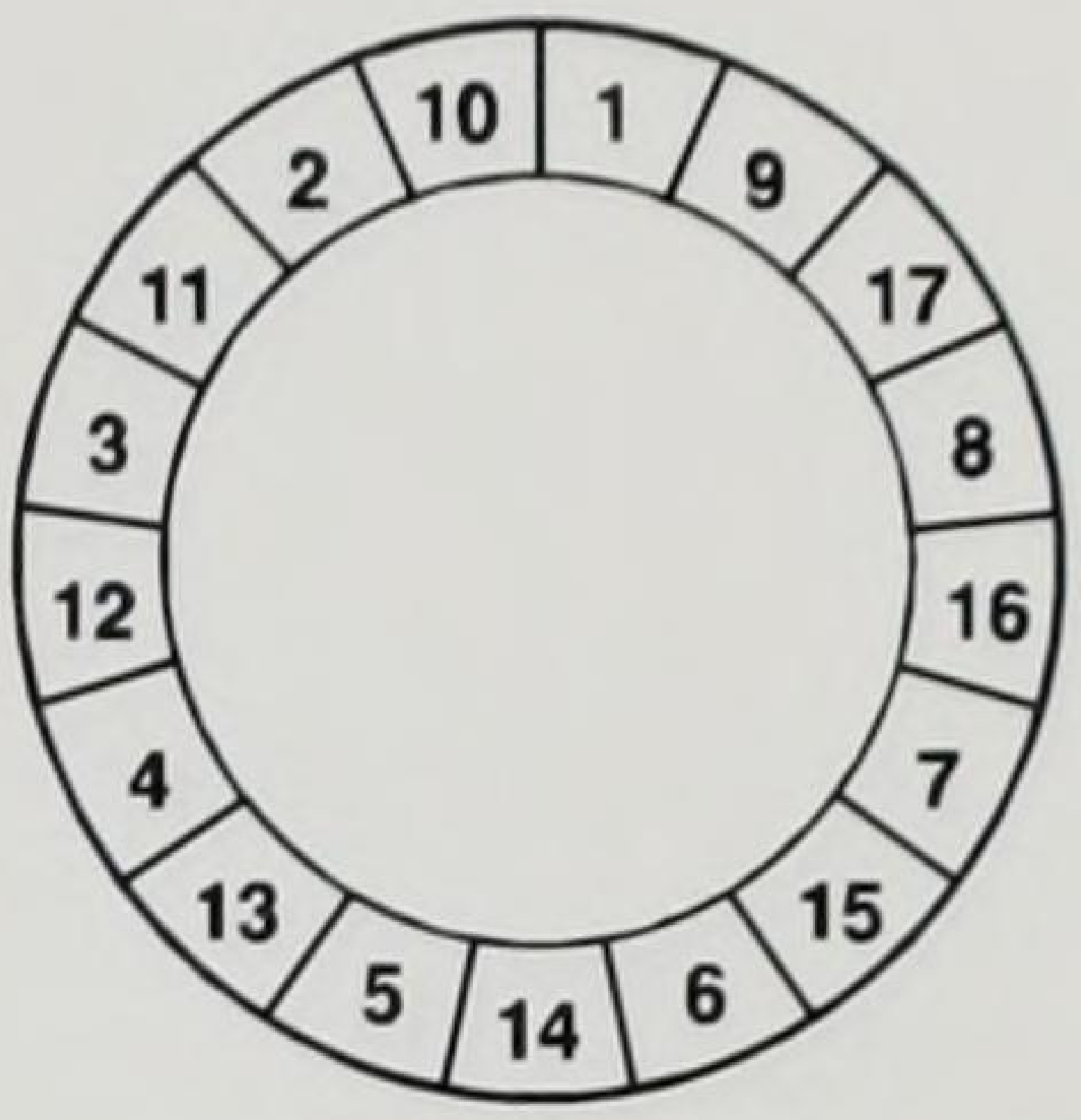
Your disk is divided into tracks, which are further divided into sectors. Depending on your disk size, the number of tracks and sectors per track will differ. By default, the sectors on each track are numbered sequentially, beginning with 1. This illustration shows tracks on a hard disk with sectors numbered 1 to 17.



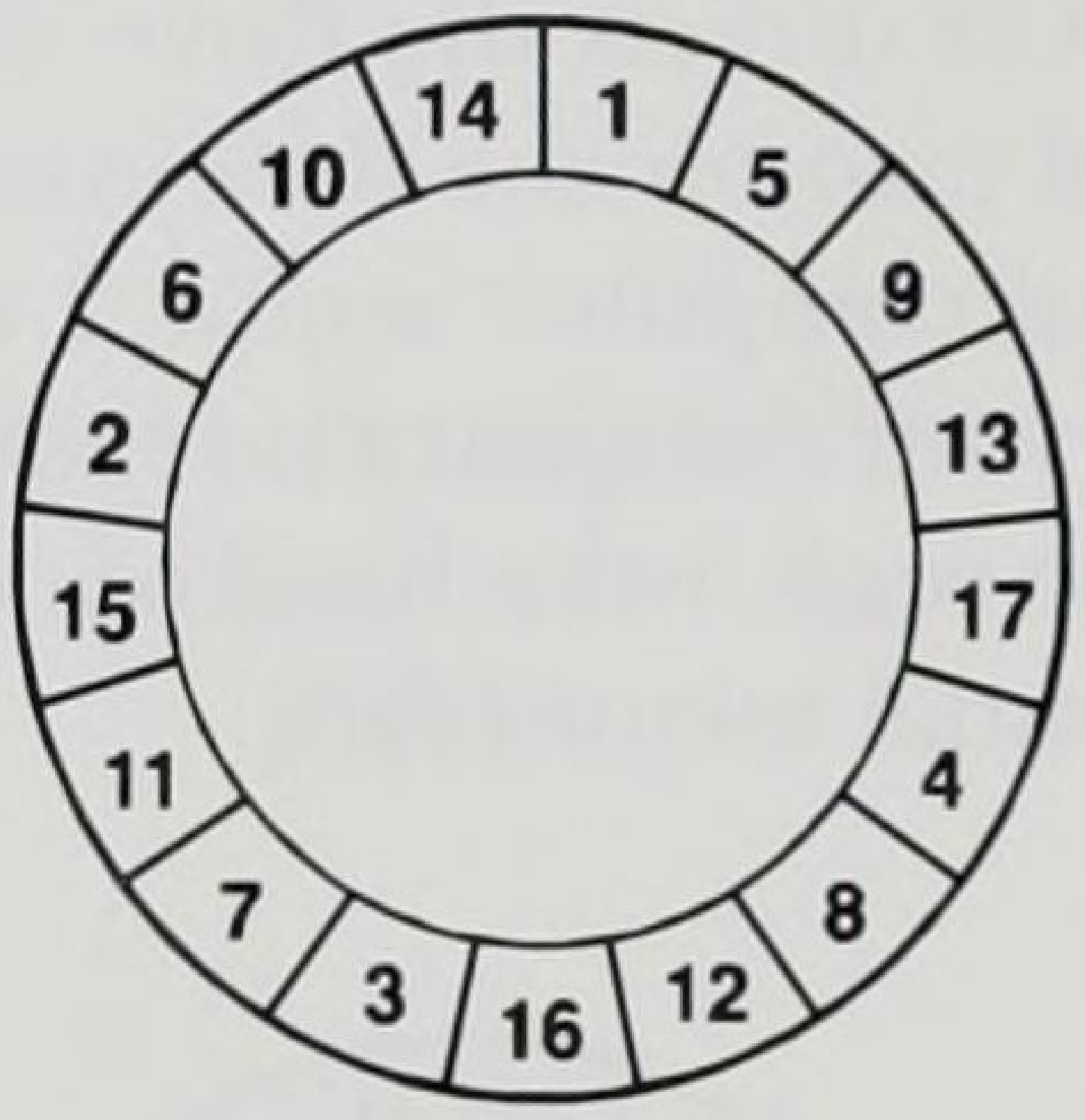
As you have learned, DOS stores your files using groups of consecutive sectors called *clusters*. Thus, when DOS reads one sector, it is very likely to immediately read the sector that follows. Assume, for example, DOS needs the information stored in sectors 1 and 2. To begin, DOS directs the drive to read sector 1. As discussed in Chapter 24, the disk drive will transfer the data into a DOS disk buffer. Depending on the speed of your disk and computer, sector 2 may have already spun past the disk's read/write head before your computer is ready for it. As a result, you must wait for the sector to rotate past the read/write head once again—your computer may be unable to keep up with sequential sector reads.



To correct this computer/disk timing problem, you can change the order that sector numbers appear on your disk. In general, you specify a sector distance, called an *interleave*, between successive disk sectors. The following illustration shows a disk with a 2:1 (2 to 1) sector interleave, meaning sector 2 appears two sector positions from sector 1 and two sector positions from sector 3. By providing the interleave, you give your computer time to process the first sector before the second sector spins past the disk's read/write head.



Likewise, the following illustration shows a 4:1 disk interleave, which might be needed for a slower computer.



If your computer is very fast, a 1:1 interleave will provide the best performance.

DOS, unfortunately, does not provide a way of determining or setting your disk's ideal interleave. To do so, you will need a special third-party software program. If you are running database programs or using large spreadsheets that make extensive use of your disk, having the correct disk interleave will make a considerable performance improvement.

Low-Level Formatting

For simplicity, we have stated the FORMAT command prepares your disk for use by dividing the disk into tracks and sectors. For floppy disks, FORMAT performs this processing. For hard disks, however, a different program, called a *low-level format*, actually divides your disks into tracks and sectors. Most users don't know about low-level formats, however, because manufacturers typically perform the low-level format for you before they ship your computer.

The low-level format operation is actually performed by the disk controller that will eventually use the disk. If you purchase a new hard disk by mail order, for example, you may need to perform a low-level format of the disk yourself.

The low-level format performs three essential functions. First, it divides your disks into tracks and sectors. Second, the low-level format assigns your disk's sector interleave. Third, it lets you record bad disk locations, which the disk manufacturer identified when testing the disk. Typically, these bad blocks are recorded on a certificate attached to the outside of your hard disk's casing.

If you purchase a new hard disk and need to perform a low-level format yourself, you should receive documentation with your disk that explains the steps to follow. In some cases, you will receive a floppy disk containing a low-level format program. In other cases, your disk controller will be built into its read-only memory (ROM), which you can access through DEBUG.

Do not perform a low-level format on a disk containing information you want to retain. The low-level format will destroy the disk's contents.

NOTE

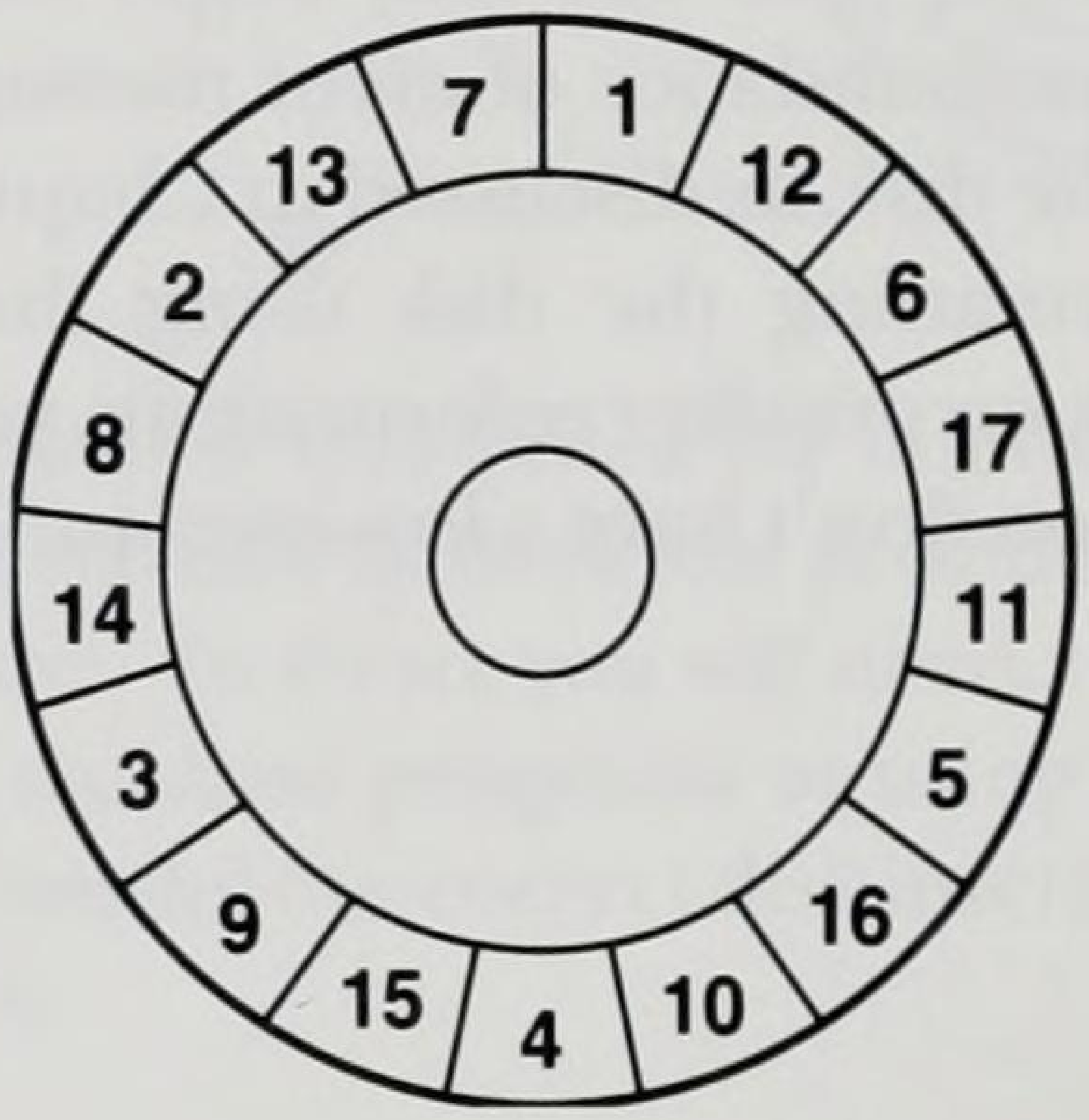
Because most users have hard disks that are already installed, low-level formatted, formatted for use by DOS, and containing files, you may be wondering why you need to be aware of low-level formatting. Over time, your disk may begin to experience disk read or write errors, as evidenced by an abundance of error messages. In such an event, you should immediately back up the disk, as discussed in Chapter 22. Next, your first corrective step should be reformatting the disk using the high-level FORMAT. If you continue to experience disk errors after reformatting, you may then need to try a low-level format operation. If you don't have a low-level program, many third-party software utilities provide one.

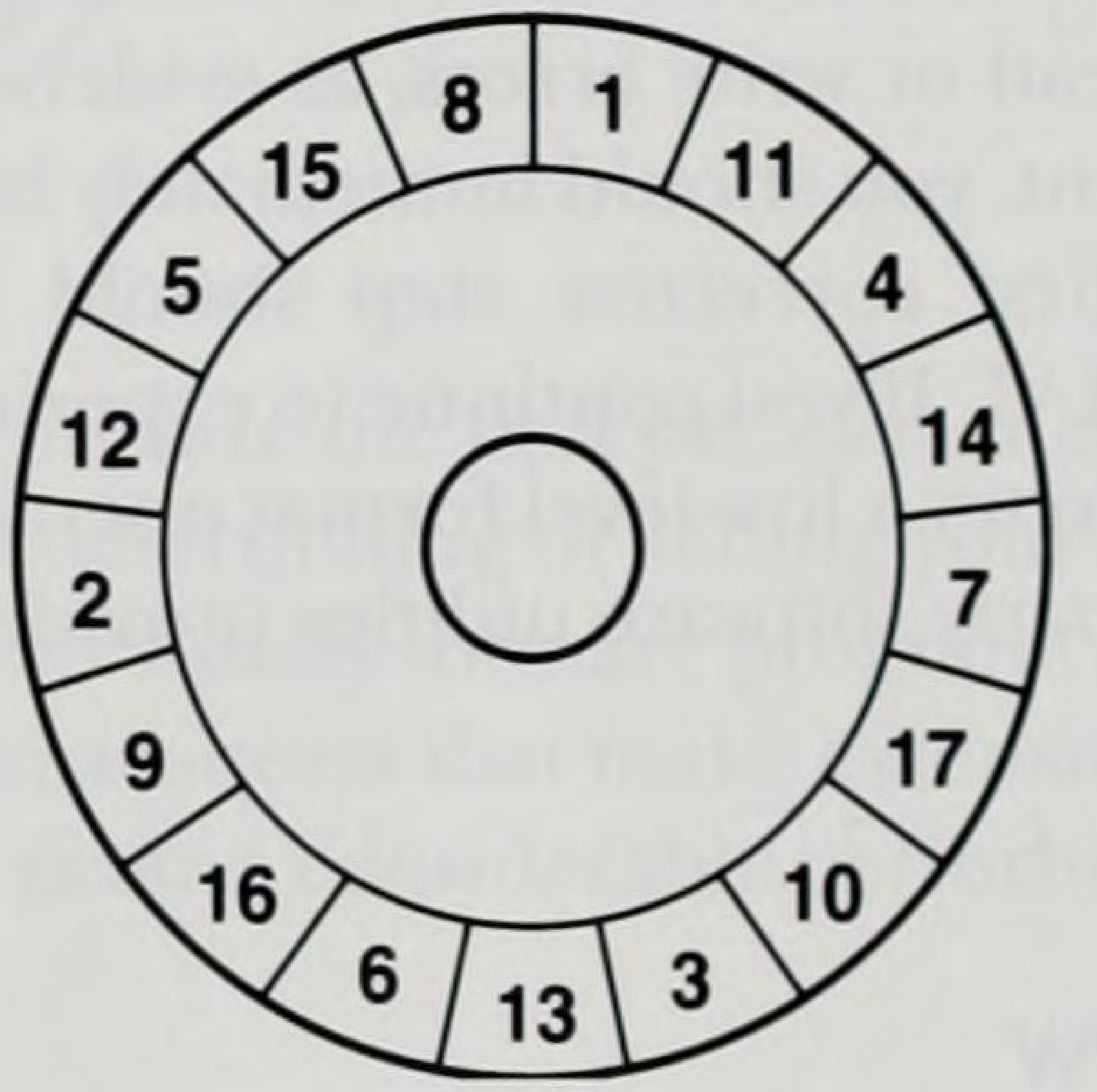
Review

1. What is disk head parking?
2. What is a disk head crash?
3. What is disk interleaving?
4. Using a 17-sector-per-track disk, draw a 3:1 disk interleave and a 5:1 disk interleave.
5. How does a low-level format operation differ from FORMAT.COM?

Answers

1. Disk head parking is the process of moving (seeking) the disk drive's read/write heads to a specific cylinder, typically the disk's innermost cylinder, to reduce the chance of the heads coming in contact with cylinders containing data. As a general rule, you should park your disk heads before you turn off your computer.
2. A disk head crash occurs when your disk drive's read/write head comes in contact with the disk's surface, which is spinning at 3600 RPMs. When a disk head crash occurs, the read/write head will scrape off the disk's specially coated surface, destroying the information the disk contained. Disk head crashes occur most often when your computer is moved while it is on with the disks spinning.
3. Disk interleaving is the physical separation of sequentially numbered disk sectors on a hard disk to compensate for timing problems between the computer and disk. Using a 2:1 interleave, for example, sector 2 is placed two sectors from sectors 1 and 3. Likewise, sector 3 is placed two sectors from sector 4, and so on.
4. The 3:1 and 5:1 disk interleaves appear as follows:





5. A low-level format performs a physical format of your hard disk, dividing the disk's surface into tracks and sectors recognizable by the disk controller. The low-level format also performs an analysis of the disk's surface to determine which areas are unusable. Using the information recorded on the disk by the low-level format, the FORMAT command performs a high-level or logical format of your hard disk, creating the structures DOS needs to store information on your disk, such as the boot record, file allocation tables, and root directory.

Advanced Concepts

Although the disk is divided into tracks and sectors, the disk controller still needs a way to identify specific sectors. To do so, the controller places header information before each sector that contains the sector's disk head (side) number, cylinder or track number, and sector address. In addition, this header can hold a sector-damaged flag stating the sector is unusable. The header ends with a CRC (cyclic redundancy check) value for error detection. The header information is followed by a small gap, which gives the controller time for the transition from reading to writing, and then the data. Following the data, the disk controller records error-corrective coding, information called ECC, that the controller can use, if necessary, to rebuild the sector's information when a disk error occurs. This information is followed by a small gap, which lets the controller change back to reading, then the following sector's header, as shown in Figure 35-3. The disk changes from reading to writing in this way because the only time sector headers are written to is during low-level formatting.

Key Terms

Cyclic redundancy check (CRC) An error-detection protocol used when recording information on a disk.

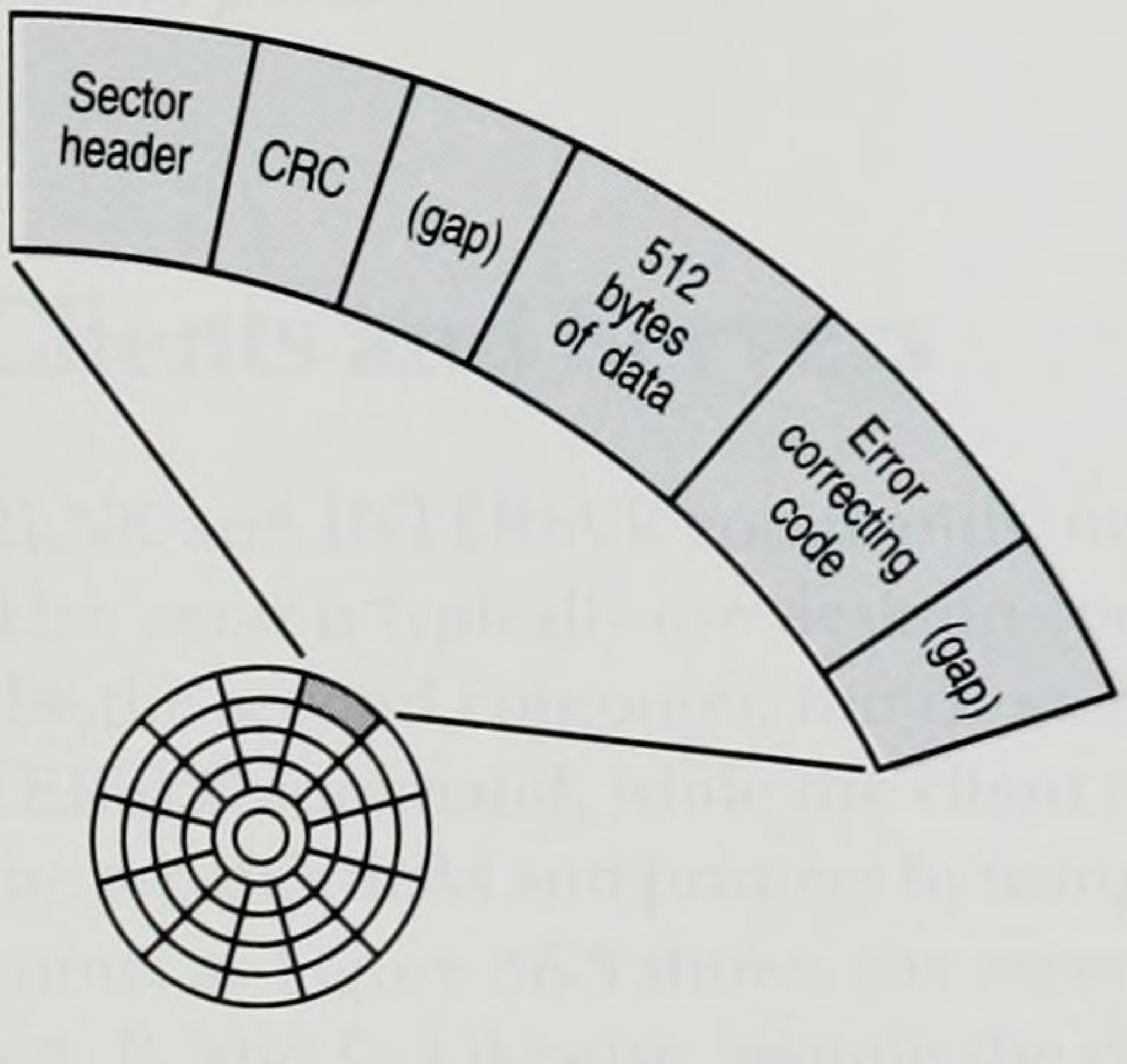


Figure 35-3. The disk controller uses header information to locate sectors on your disk

Disk parking The process of moving (often called seeking) the disk's read/write head to the innermost disk cylinder to reduce the chance of the disk head coming in contact with tracks containing data.

Disk crash A problem occurring when the disk drive's read/write head comes in contact with the disk's rotating surface, scraping the disk surface and destroying the information it contains.

Sector interleaving The process of separating sequentially numbered sectors to reduce sector misses when the computer can't keep up with the disk drive.

Low-level formatting The physical formatting of the hard disk performed by the firmware or hard disk controller.

High-level formatting The logical formatting of a disk by FORMAT to build the file allocation table and root directory.

Chapter 36

Linking Computers with DOS 6

Over the past few years, sales for laptop and notebook computers have grown at a tremendous rate. Many users now take a notebook computer with them everywhere they go. Upon returning to their home or office, the users often have to shuffle many floppies between their desktop and notebook computers to bring each system's hard disks up to date. To make information exchange easier between two systems, DOS 6 provides the INTERLNK and INTERSVR commands. By using these commands and connecting the computers as shown in Figure 36-1, two systems can quickly exchange files or share a disk or printer. The connection can be made either with a bidirectional parallel cable between two parallel ports, or with a *NULL-modem cable* (which replaces a modem) between two serial ports.

Understanding Clients and Servers

With the DOS 6 INTERLNK and INTERSVR commands, one system is a server and the second is a client. The *server* is typically the desktop computer whose disks and printers can be shared by the second computer, the *client*. As shown in Figure 36-2, the server runs the INTERSVR command, while the client runs INTERLNK.

The client accesses the server's disks and printers by using a drive letter or printer name. For example, assume, as Figure 36-3 shows, the server has a printer attached to LPT1, and the drives A, B, and C. Likewise, assume the client has drives A, B, C.

After you connect the computers, as shown in Figure 36-4, the client would access the server's disks using the drive letters D, E, and F. Likewise, the client would access the server's printer using the device name LPT2.

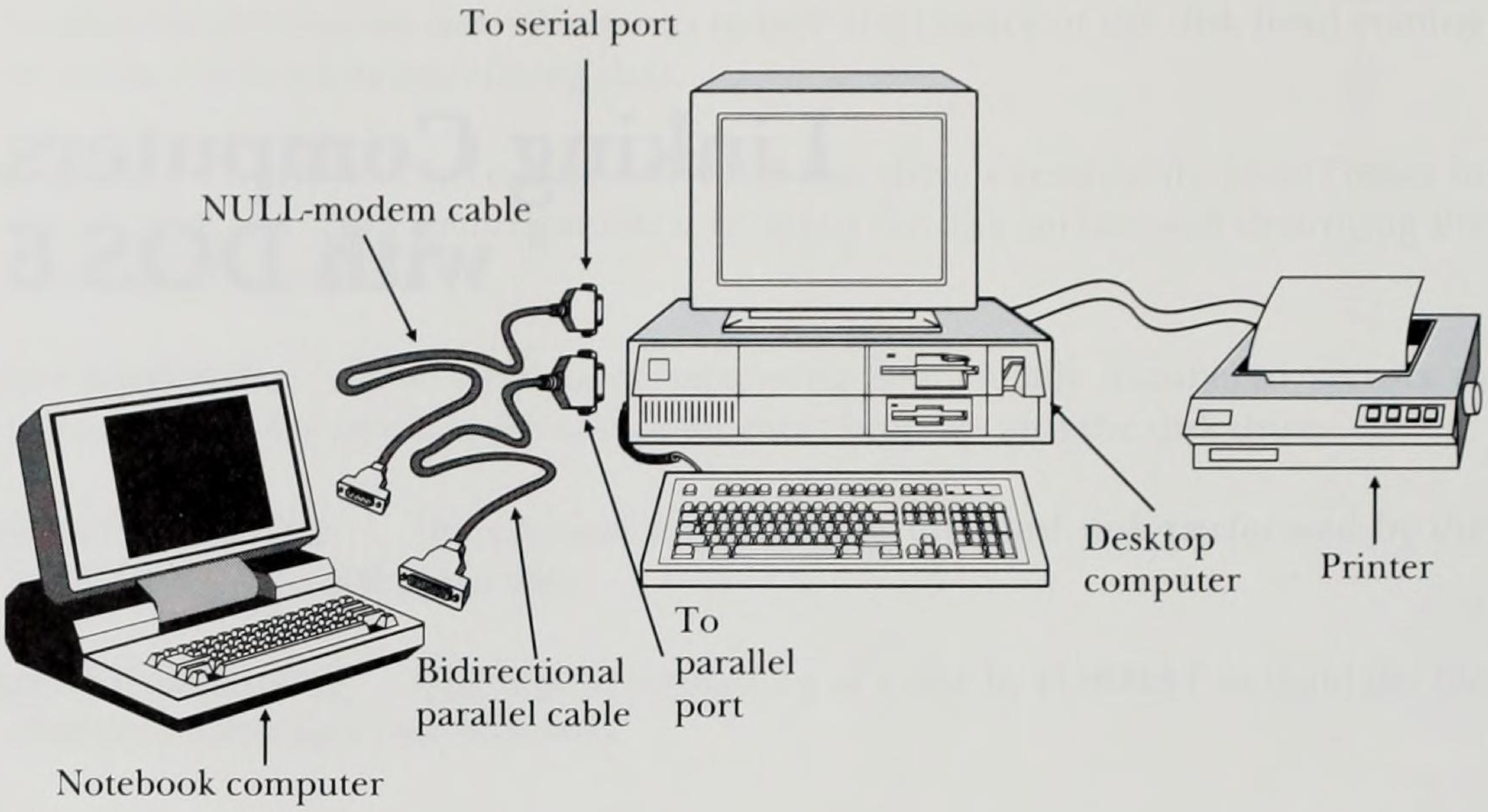

To serial port
NULL-modem cable
Desktop computer
Printer
To parallel port
Bidirectional parallel cable
Notebook computer

Figure 36-1. *Connecting computers for file exchange*

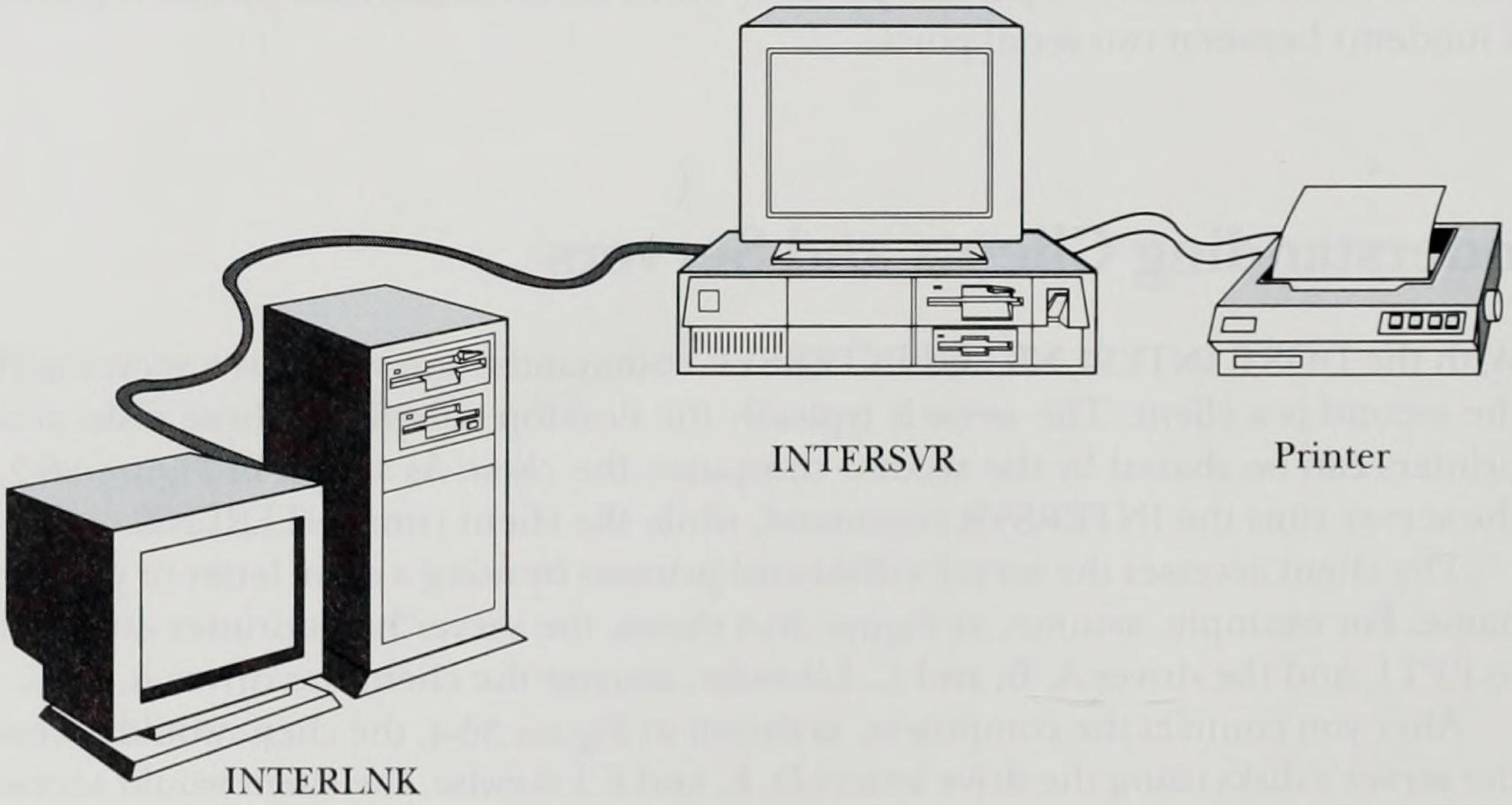

INTERSVR
Printer
INTERLNK

Figure 36-2. *The server runs INTERSVR while the client runs INTERLNK*

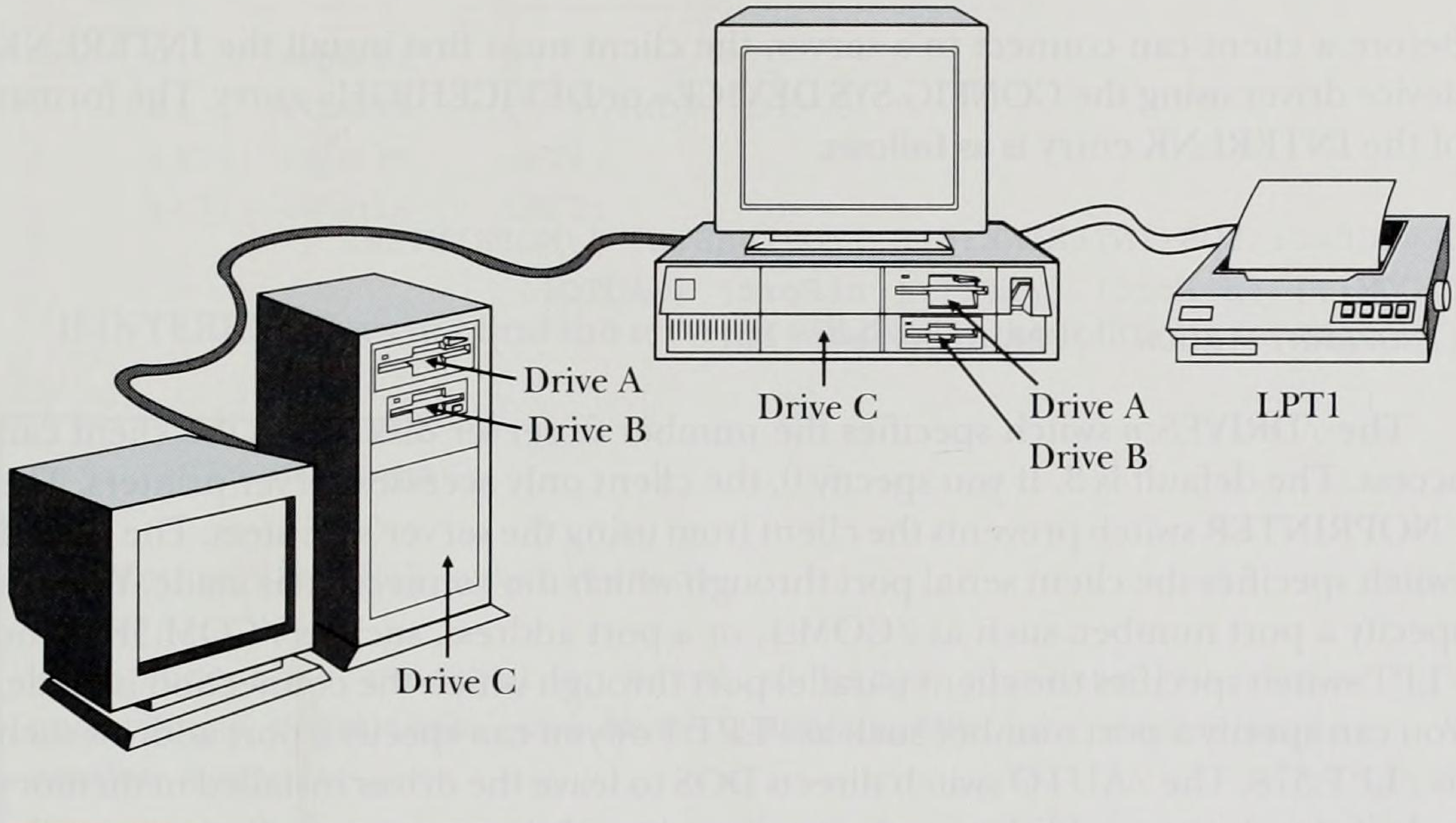

Drive A
Drive B
Drive C
Drive C
Drive A
Drive B
LPT1

Figure 36-3. *Server and client disk drive and printer names*

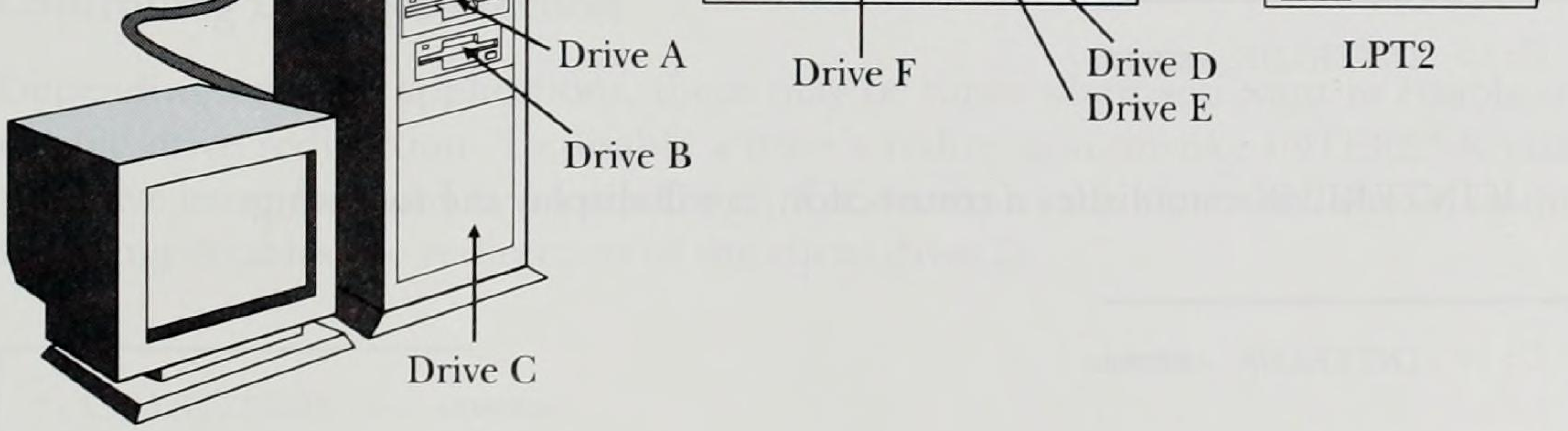

Drive A
Drive B
Drive C
Drive F
Drive D
Drive E
LPT2

Figure 36-4. *Client remote device names*

Installing the INTERLNK Device Driver

Before a client can connect to a server, the client must first install the INTERLNK device driver using the CONFIG.SYS DEVICE= or DEVICEHIGH= entry. The format of the INTERLNK entry is as follows:

```
DEVICE=C:\DOS\INTERLNK.EXE [/DRIVES:n] [/NOPRINTER]
[/COM[:] [n|Port] [/LPT[:][n|Port] [/AUTO]
[/NOSCAN] [/LOW] [/BAUD:Rate] [/V]
```

The /DRIVES:*n* switch specifies the number of server disk drives the client can access. The default is 3. If you specify 0, the client only accesses server printers. The /NOPRINTER switch prevents the client from using the server's printers. The /COM switch specifies the client serial port through which the connection is made. You can specify a port number, such as /COM:1, or a port address, such as /COM:3F8. The /LPT switch specifies the client parallel port through which the connection is made. You can specify a port number such as /LPT:1 or you can specify a port address such as /LPT:378. The /AUTO switch directs DOS to leave the driver installed in memory only if the driver establishes a connection when the system starts. In contrast, the /NOSCAN switch loads the driver, directing the driver not to establish a connection during startup. The /LOW switch loads the driver into conventional memory. By default, the driver uses the upper memory area. The /BAUD switch specifies the communication speed for serial port connections. Valid rates include 9600, 19200, 38400, 57600, and 115200. The default baud rate is 115200. Finally, the /V switch reduces possible PC timer conflicts during serial port connections. Use /V if one of the systems hangs up during file or printer operations.

The following DEVICE= entry loads the INTERLNK driver for serial communication through COM1:

```
DEVICE=C:\DOS\INTERLNK.EXE  /COM:1
```

After you place the entry in CONFIG.SYS and restart your system, INTERLNK is ready for use. Assuming your computer is connected to a server that is running INTERSVR, you can establish a connection by invoking INTERLNK as follows:

```
C:\> INTERLNK <ENTER>
```

If INTERLNK establishes a connection, it will display the following:

```
C:\> INTERLNK <ENTER>

    This Computer         Other Computer
```

```
     (Client)           (Server)
   -------------    -------------------------

     D:    equals    A:
     E:    equals    C: (61Mb)  DOS 6
     LPT2: equals    LPT1:
     LPT3: equals    LPT2:
```

If INTERLNK does not find the server, it will display the following message:

```
C:\> INTERLNK <ENTER>

    Connection NOT established

Make sure that a serial or parallel cable connects the server
and client computers, and that INTERSVR.EXE is running on the
server computer.
```

If this message appears, follow these trouble-shooting steps:

- Is the server computer running INTERSVR?
- Is the cable properly connected?
- Are you sure the cable is a NULL-modem cable for serial port connections or bidirectional cable for parallel port connections?
- Is the cable connected to the port you specified in the INTERLNK and INTERSVR commands?
- Are the client and server using the same baud rate for serial communication?

Controlling Drive Redirection

Depending on your applications, there may be times when you want to enable or disable drive redirection. To disable a drive's redirection, invoke INTERLNK with the drive letter followed by an equal sign. For example, the command shown in the following disables the redirection of the client drive D.

```
C:\> INTERLNK D= <ENTER>
```

If you later want to redirect the client drive, invoke INTERLNK with the client and server drives as shown here:

```
C:\> INTERLNK  D:=A: <ENTER>
```

Using INTERSVR to Create a Server

To establish a client/server connection, the client computer must be running INTERLNK and the server must be running INTERSVR. The format of INTERSVR is as follows:

```
INTERSVR [Drive:] [/X:Drive] [/COM[:][n|Port]
[/LPT[:][n|Port] [/Baud:Rate] [/B] [/V]
```

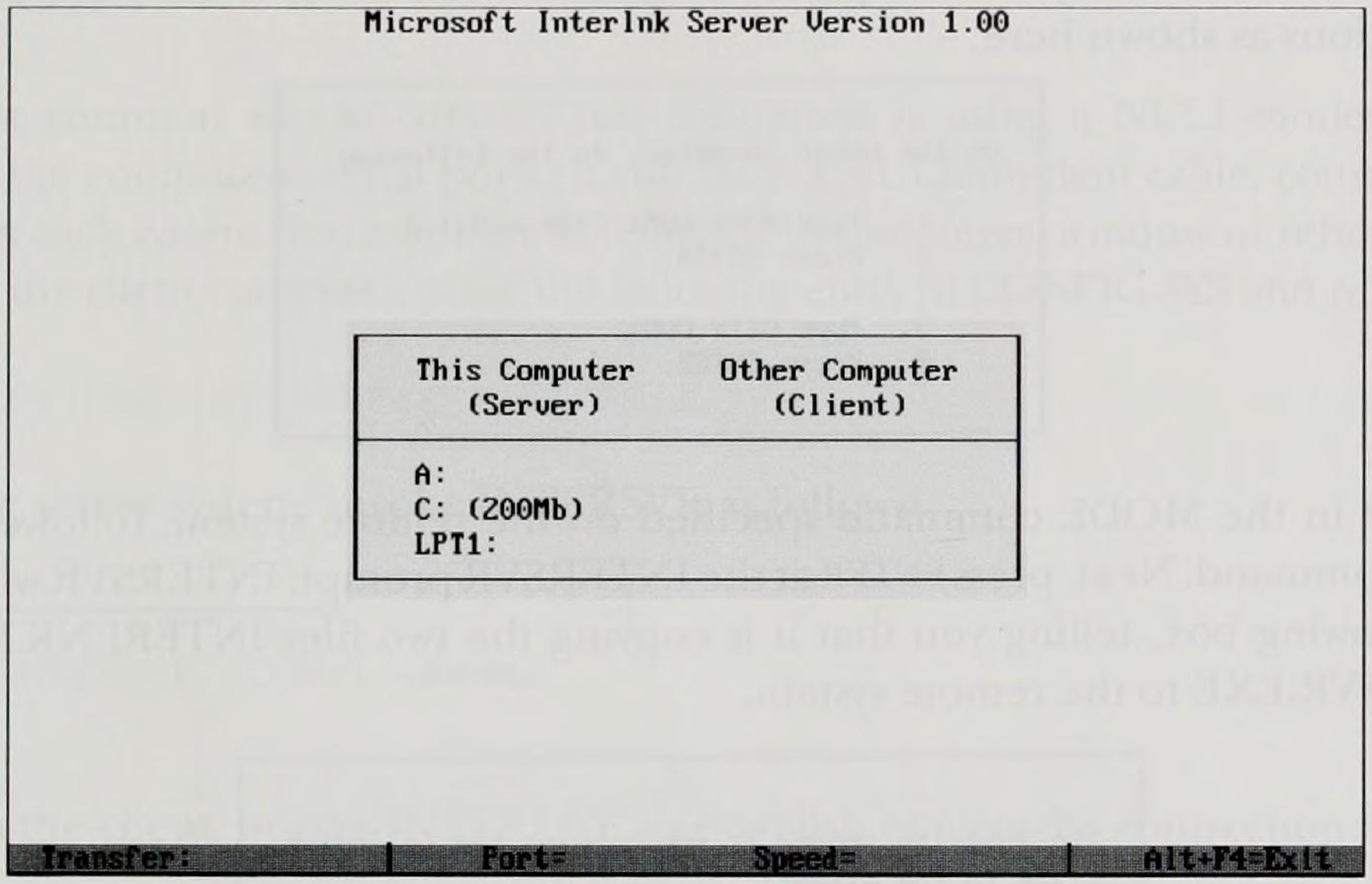
If you specify a drive, INTERSVR will only allow the client to access that specific drive. By default, INTERSVR allows the client to access all drives. The /X switch lets you exclude access to a specific drive. For example, if you want the client to access all drives but drive C, you would use /X:C. The /COM switch specifies the server serial port through which the connection will be made. You can specify a port number such as /COM:1 or a port address such as /COM:3F8. The /LPT switch specifies the server parallel port through which the connection will be made. You can specify a port number as /LPT:1 or you can specify a port address such as /LPT:378. The /BAUD switch specifies the communication speed for serial port connections. Valid rates include 9600, 19200, 38400, 57600, and 115200. The /B switch directs the server to display its screen output in black and white. Finally, the /V switch reduces possible PC timer conflicts during serial port connections. Use /V if one of the systems hangs up during file or printer operations.

Unlike INTERLNK, which requires a device driver, you simply invoke INTERSVR from the DOS prompt. The following command invokes INTERSVR for a serial connection using COM1:

```
C:\> INTERSVR /COM:1 <ENTER>
```

INTERSVR will display the screen shown in Figure 36-5.

When INTERSVR is running, you cannot run other commands on the server. After the client has finished its file or printer operations, you can end INTERSVR by pressing the ALT-F4 keyboard combination.



```
Microsoft Interlnk Server Version 1.00

 This Computer      Other Computer
   (Server)            (Client)

 A:
 C: (200Mb)
 LPT1:
```

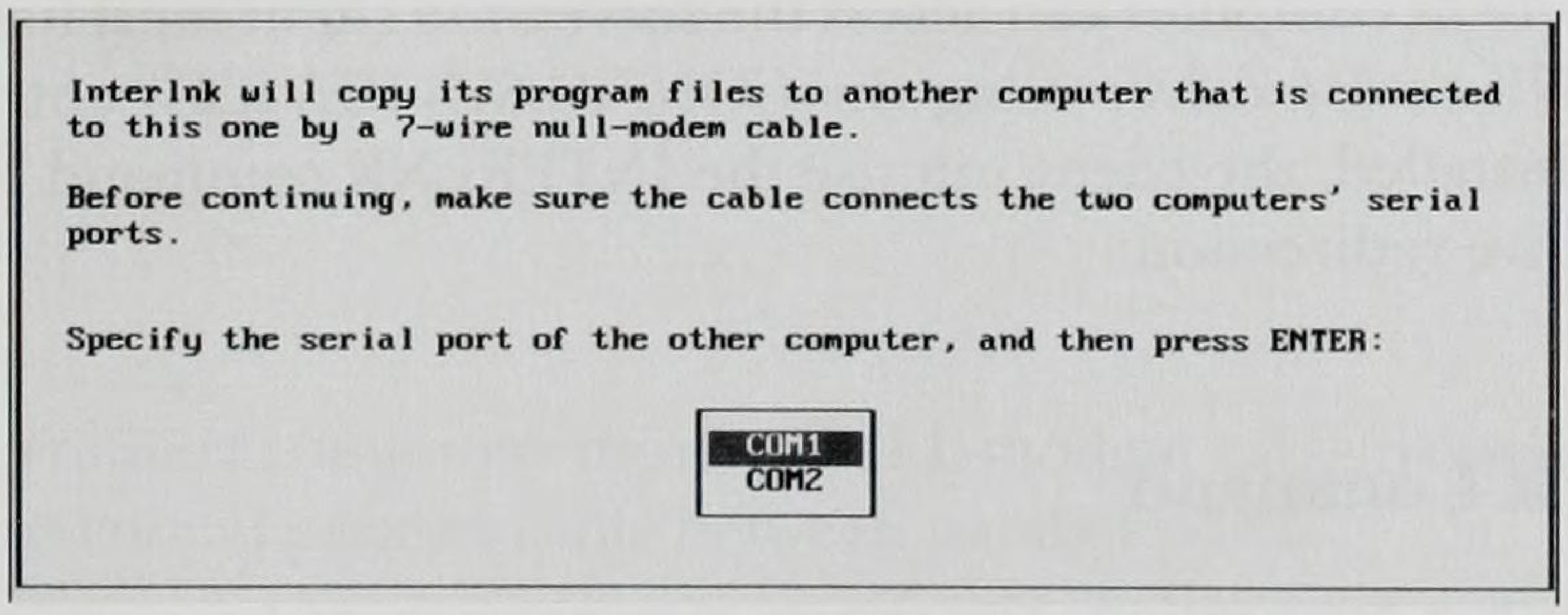
Figure 36-5. *The INTERSVR connection screen*

Copying the INTERLNK and INTERSVR Files

If your client or server system does not contain the INTERLNK or INTERSVR files, you can use INTERSVR to copy the files from one system to another. To do so, you must connect the computers using a seven-wire NULL-modem cable between serial ports. Next, assuming you have connected the computers using COM1, invoke INTERSVR from the system containing the files, as shown here:

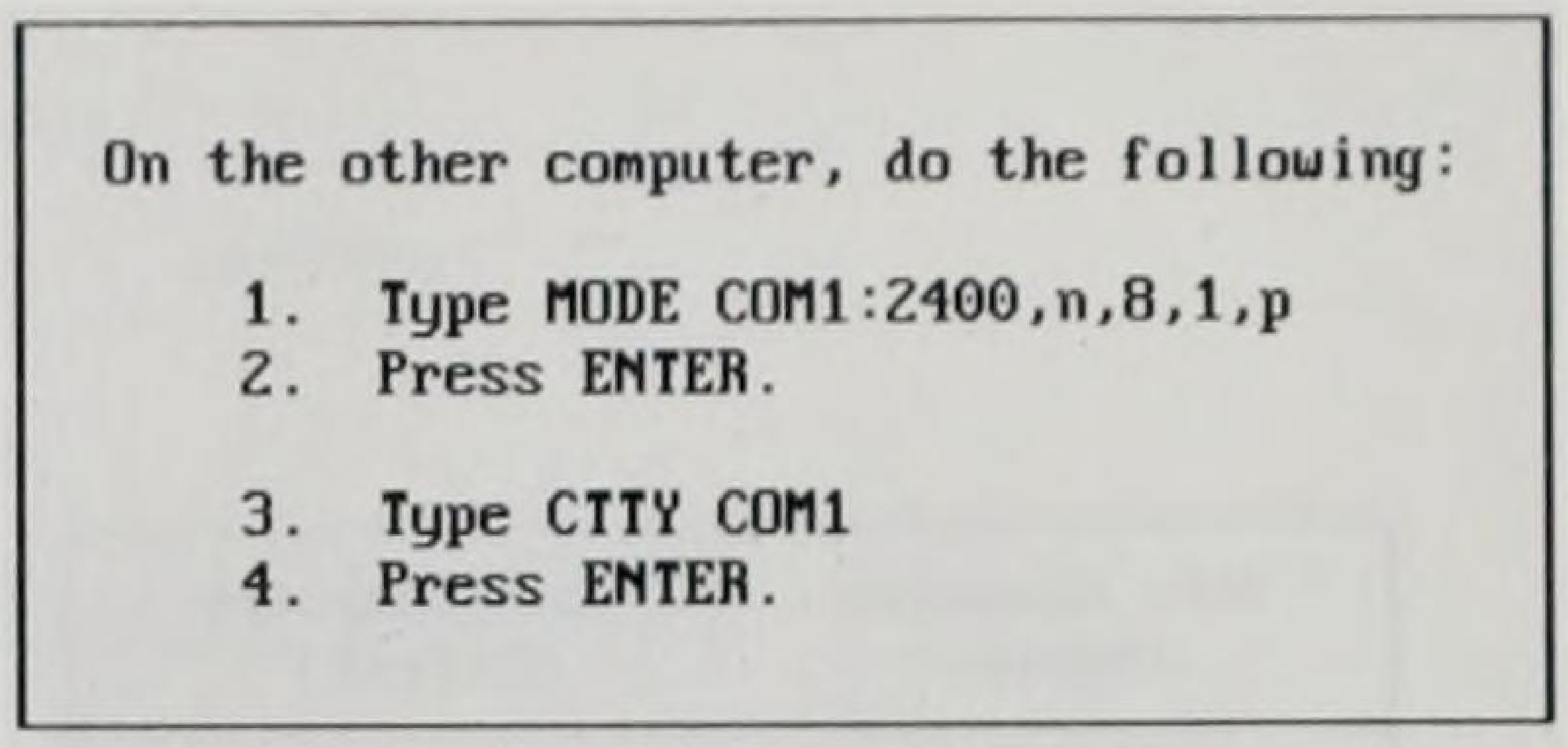
```
C:\> INTERSVR /COM:1 /RCOPY <ENTER>
```

INTERSVR will display the following box, which asks you to specify the port on the remote system through which the connection is being made.



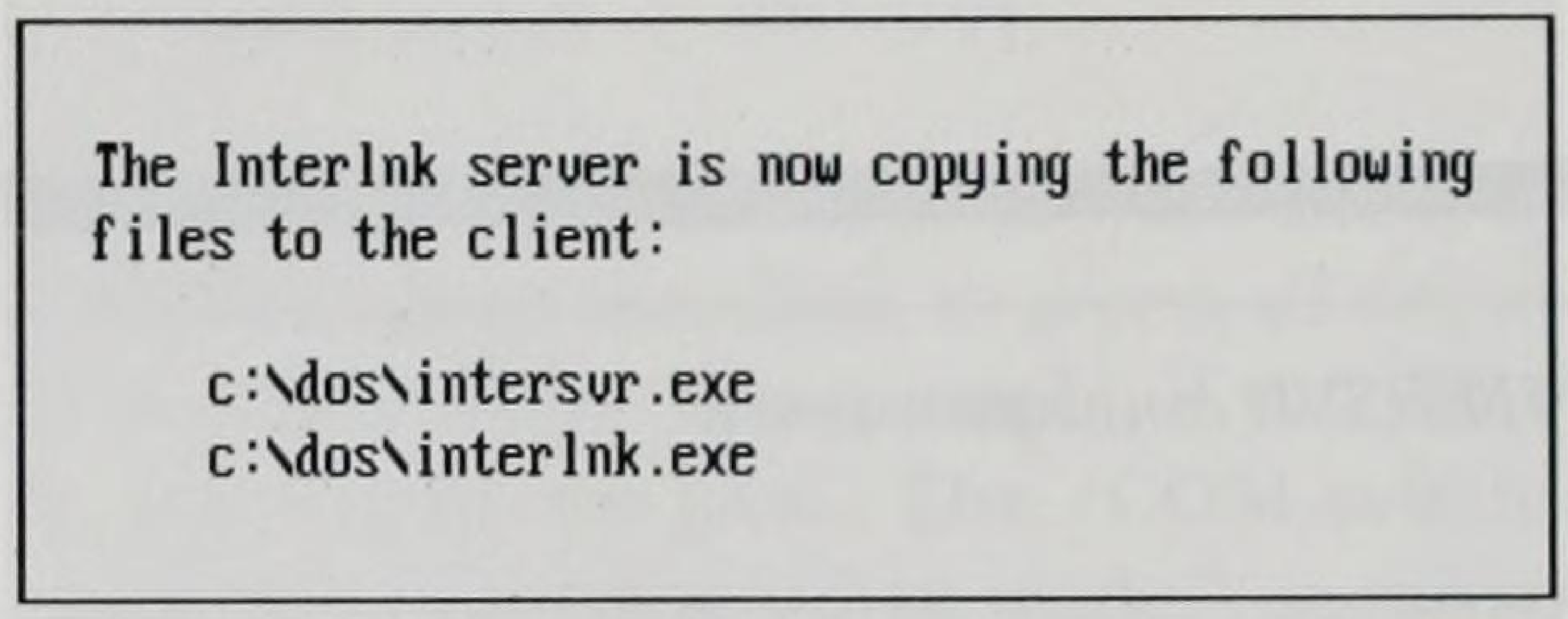
```
Interlnk will copy its program files to another computer that is connected
to this one by a 7-wire null-modem cable.

Before continuing, make sure the cable connects the two computers' serial
ports.


Specify the serial port of the other computer, and then press ENTER:
```

Select the desired port and press ENTER. INTERSVR will display a screenful of instructions as shown here.



```
On the other computer, do the following:

    1.  Type MODE COM1:2400,n,8,1,p
    2.  Press ENTER.

    3.  Type CTTY COM1
    4.  Press ENTER.
```

Type in the MODE command specified on the remote system, followed by the CTTY command. Next, press ENTER at the INTERSVR prompt. INTERSVR will display the following box, telling you that it is copying the two files INTERLNK.EXE and INTERSVR.EXE to the remote system.



```
The Interlnk server is now copying the following
files to the client:

    c:\dos\intersvr.exe
    c:\dos\interlnk.exe
```

When the copy operation completes, INTERSVR will redisplay the DOS prompt.

SUMMARY

Understanding Clients and Servers

When you connect two computers, one computer is called the server and the second is called the client. The server computer contains disks and printers that it allows the client computer to use. The server computer runs the DOS 6 INTERSVR command while the client runs INTERLNK.

INTERLNK Device Driver

Before a client computer can access the server, the client must first install the INTERLNK device driver using the CONFIG.SYS DEVICE= entry. After the driver is installed, the client can use the INTERLNK command to enable or disable drive redirection.

INTERSVR Command

To establish a connection, the server computer must be running the command INTERSVR. When INTERSVR is active, you cannot issue other commands at the server.

Hands On

The most common way to connect two computers is using a NULL-modem cable between the computers' serial ports. If you have a NULL-modem cable, connect it to COM1 on each system. You may have to temporarily disconnect a mouse or other device. Next, on the client computer, place the following entry in CONFIG.SYS and reboot:

```
DEVICE=C:\DOS\INTERLNK.EXE  /COM:1
```

At the server system, invoke INTERSVR as follows:

```
C:\> INTERSVR /COM:1 <ENTER>
```

From the client, invoke INTERLNK to establish or view the connection.

```
C:\> INTERLNK <ENTER>
```

When the connection is established, use DIR to view files on the server disk. Use PRINT to print a file to the remote printer. To end the connection, press ALT-F4 on the server to end INTERSVR.

Review

1. What are the two ways to connect two computers?
2. What is the CONFIG.SYS entry to load the INTERLNK driver for a connection over COM2?
3. What is the INTERSVR command to establish a connection over COM2?
4. When would you use the INTERSVR /RCOPY switch?

Answers

1. You can connect computers using a NULL-modem cable between serial ports or a bidirectional parallel cable between parallel ports.
2. The following CONFIG.SYS DEVICE= entry installs the INTERLNK driver for a connection using COM2:

```
DEVICE=C:\DOS\INTERLNK.EXE  /COM:2
```

3. The following INTERSVR command directs the server to use COM2:

```
C:\> INTERSVR  /COM:2 <ENTER>
```

4. The INTERSVR /RCOPY switch lets you copy the files INTERSVR.EXE and INTERLNK.EXE to a remote computer.

Key Terms

Client A computer that uses the resources, such as a disk or printer, of another computer.

NULL-modem cable A cable that connects two computers by the serial ports directly, bypassing the need for a modem (thus the word "null").

Server A computer that provides services or resources that can be used by another computer.

Chapter 37

The DOS 5 and 6 Editor, EDIT

For almost a decade, EDLIN, a line editor, was the only tool DOS provided to users to create text and batch files. EDLIN is a line editor because it only lets you work with a single line of the file at a time. Starting with version 5, DOS provides EDIT, a full-screen editor. EDIT works much like a word processor, letting you view a file's contents on the screen, traversing from one position in the file to another using your keyboard's arrow and page keys.

This chapter takes a look at EDIT in detail. To begin, you will learn how to create, save, and print files with EDIT using either your mouse or keyboard. Next, you will learn how to customize EDIT's screen display and how to use EDIT's online help. In the "Advanced Concepts" section you will learn how to search a large file for key text, how to change text, and how to move text from one location in your file to another (cut and paste).

Starting EDIT

The simplest way to create a file using EDIT is to include the filename in EDIT's command line, as shown in the following example.

```
C:\> EDIT  FILENAME.EXT
```

If the file exists, EDIT will display the file's contents; if not, EDIT will display an empty editing window. Assume, for example, you want to create the simple batch file DATETIME.BAT, which contains the following commands:

```
@ECHO OFF
DATE
TIME
```

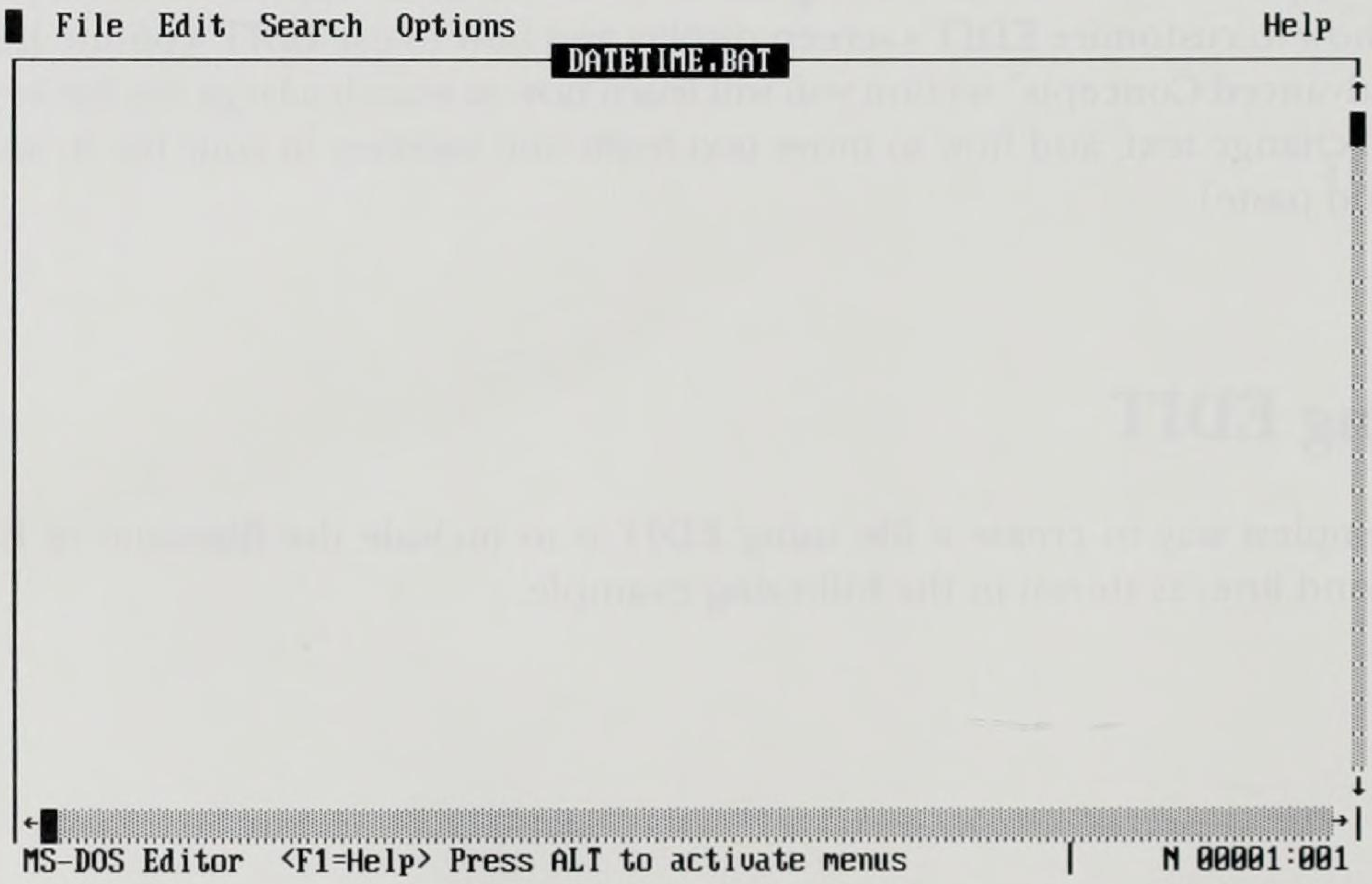
To begin, you would invoke EDIT as follows.

```
C:\> EDIT  DATETIME.BAT  <ENTER>
```

Because the file does not exist, EDIT displays an empty editing window, as shown in Figure 37-1.

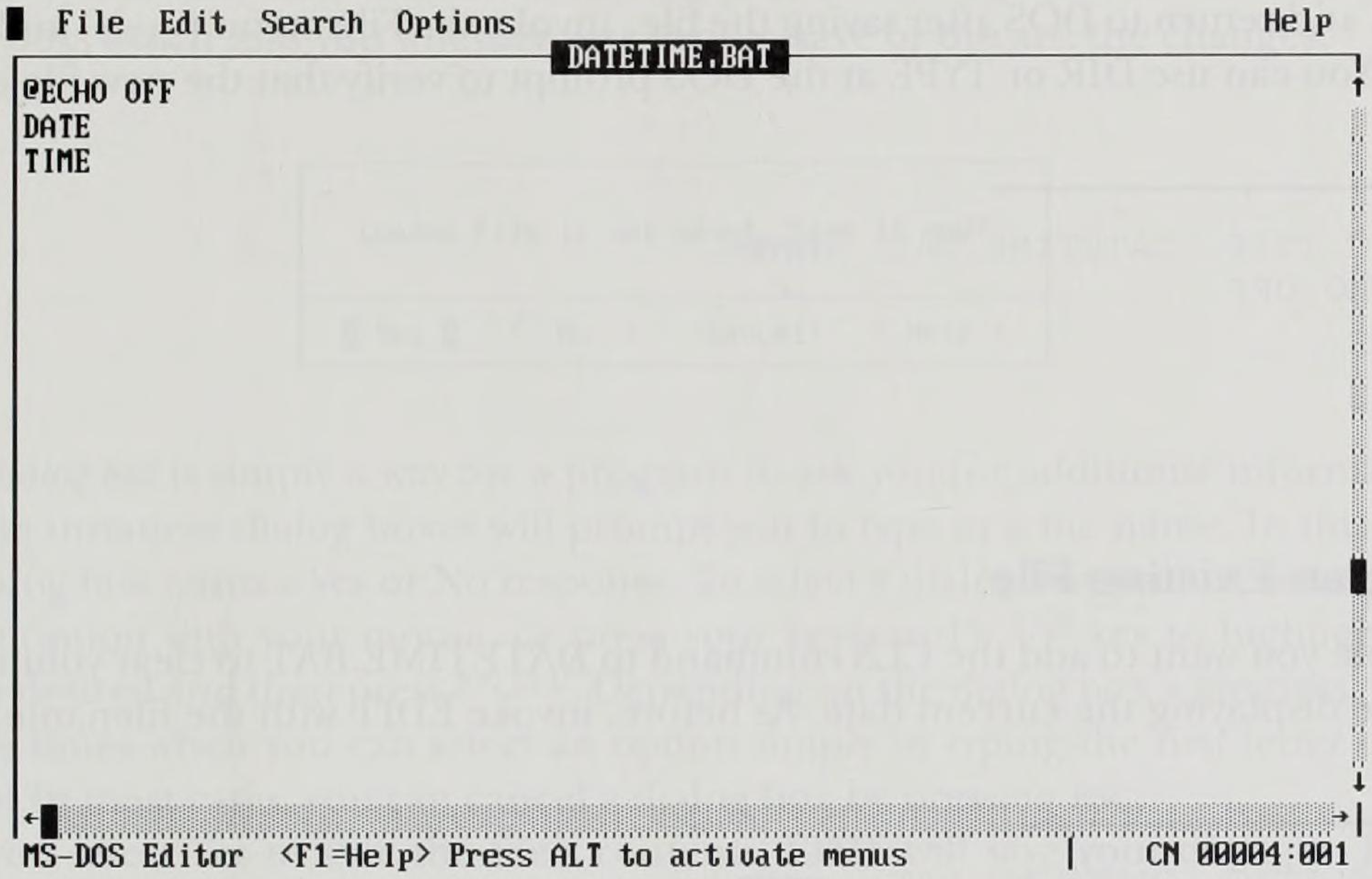
Next, simply type the batch file's contents. The lines appear in the window, as in Figure 37-2.

To save the file's contents to disk, you must invoke EDIT's File menu. If you examine the top row of your screen, you'll find the names of five menus: File, Edit,



```
MS-DOS Editor  <F1=Help> Press ALT to activate menus          |     N 00001:001
```

Figure 37-1. *EDIT's empty editing window*

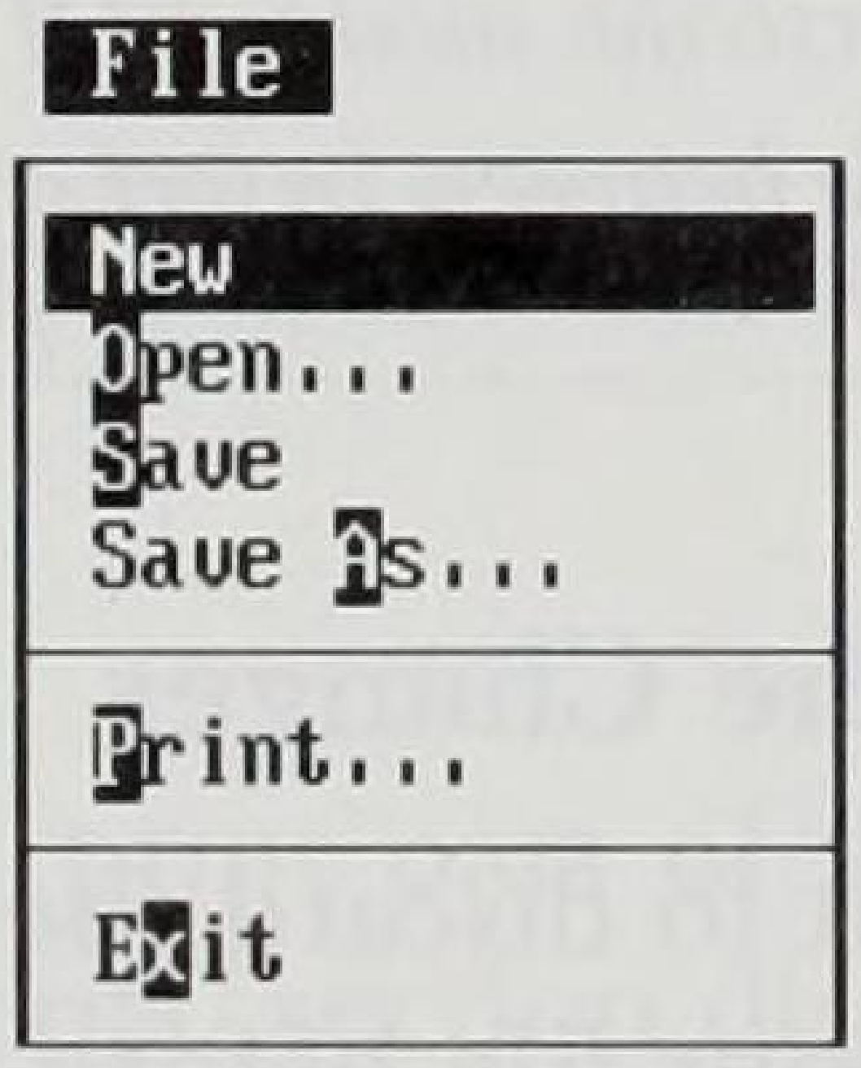

File Edit Search Options Help
DATETIME.BAT
MS-DOS Editor <F1=Help> Press ALT to activate menus
CN 00004:001

```
@ECHO OFF
DATE
TIME
```

Figure 37-2. *Editing window showing the contents of DATETIME.BAT*

Search, Options, and Help. If you hold down your keyboard's ALT key, EDIT will highlight the first letter of each menu's name. By typing one of these letters while ALT is depressed, you select the menu. To invoke EDIT's File menu, you would press ALT-F. If your computer has a mouse and you have installed the appropriate mouse device driver, EDIT will let you use your mouse to edit your file or to select a menu. To select EDIT's File menu, for example, position the mouse pointer at the word "File" and click the left button.

When you select the File menu, EDIT displays the following pull-down menu:


File
New
Open...
Save
Save As...
Print...
Exit

You can select a menu option by using your mouse to click on the option, or by pressing the key that corresponds to one of the highlighted letters.

Note the options that are followed by three periods (called ellipses). If you select one of these options, EDIT will display another menu or a prompt for additional information, such as a filename. In this case, to save the file's contents to disk, select the Save option either by clicking on the option with your mouse, by typing **S**, or by moving the menu highlight with your arrow keys to Save, and pressing ENTER. To exit EDIT and return to DOS after saving the file, invoke the File menu (ALT-F) and select Exit. You can use DIR or TYPE at the DOS prompt to verify that the new file exists.

```
C:\> TYPE  DATETIME.BAT  <ENTER>
@ECHO OFF
DATE
TIME
```

Editing an Existing File

Assume you want to add the CLS command to DATETIME.BAT to clear your screen before displaying the current date. As before, invoke EDIT with the filename.

```
C:\> EDIT  DATETIME.BAT  <ENTER>
```

EDIT will display its editing window with the file's current contents. Using your keyboard's arrow keys, move the cursor to the first character on line 2. If you look at the lower-right corner of the editing window, you will notice a series of numbers separated by a colon, such as 00002:001. The first five digits tell you the current line number. The second three numbers tell you the current column. As your files become large, you can use these numbers to keep track of your current location within them.

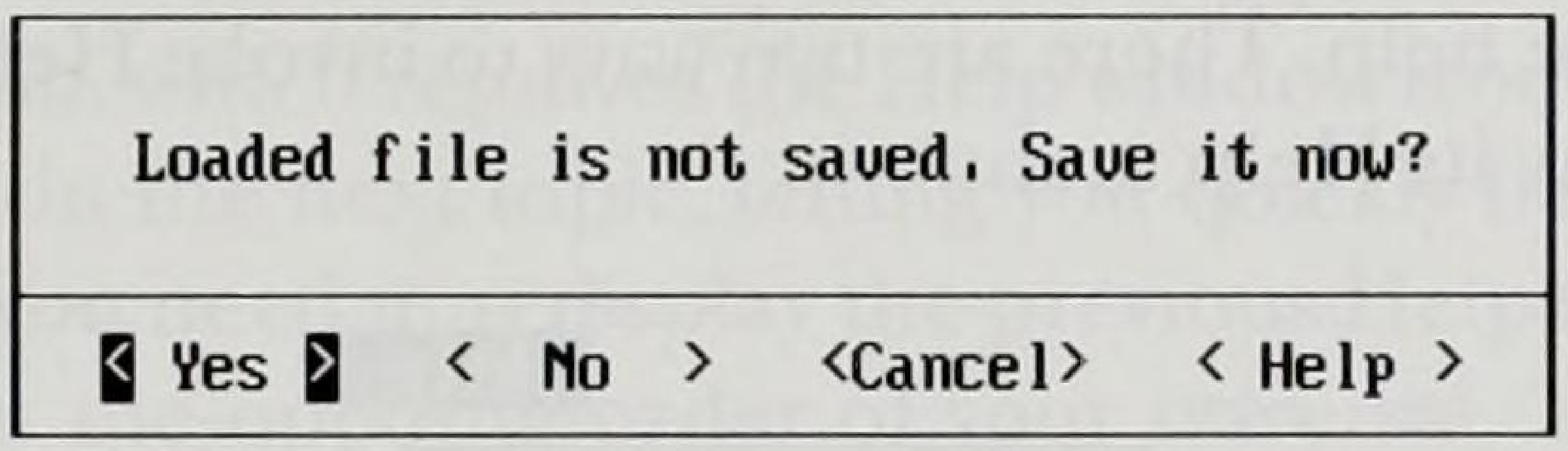
Type CLS and press ENTER. EDIT will insert the command into your batch file. Invoke the File menu's Save option to save the file's contents. Next, use the File menu's Exit option to exit EDIT and go to DOS.

NOTE *EDIT supports insert and overstrike modes. In insert mode, EDIT inserts all text you type to the left of existing text. In overstrike mode, EDIT overwrites existing text with the text you type. Use the INS key to toggle between insert and overstrike mode.*

Exiting EDIT Without Saving the Changes

There may be times when you want to discard the changes you have made to a file, or times when you forget to save your file before selecting the Exit option to leave

EDIT. If you invoke Exit without first saving your file changes, EDIT will display this dialog box, which asks you whether you want to save or discard the changes.



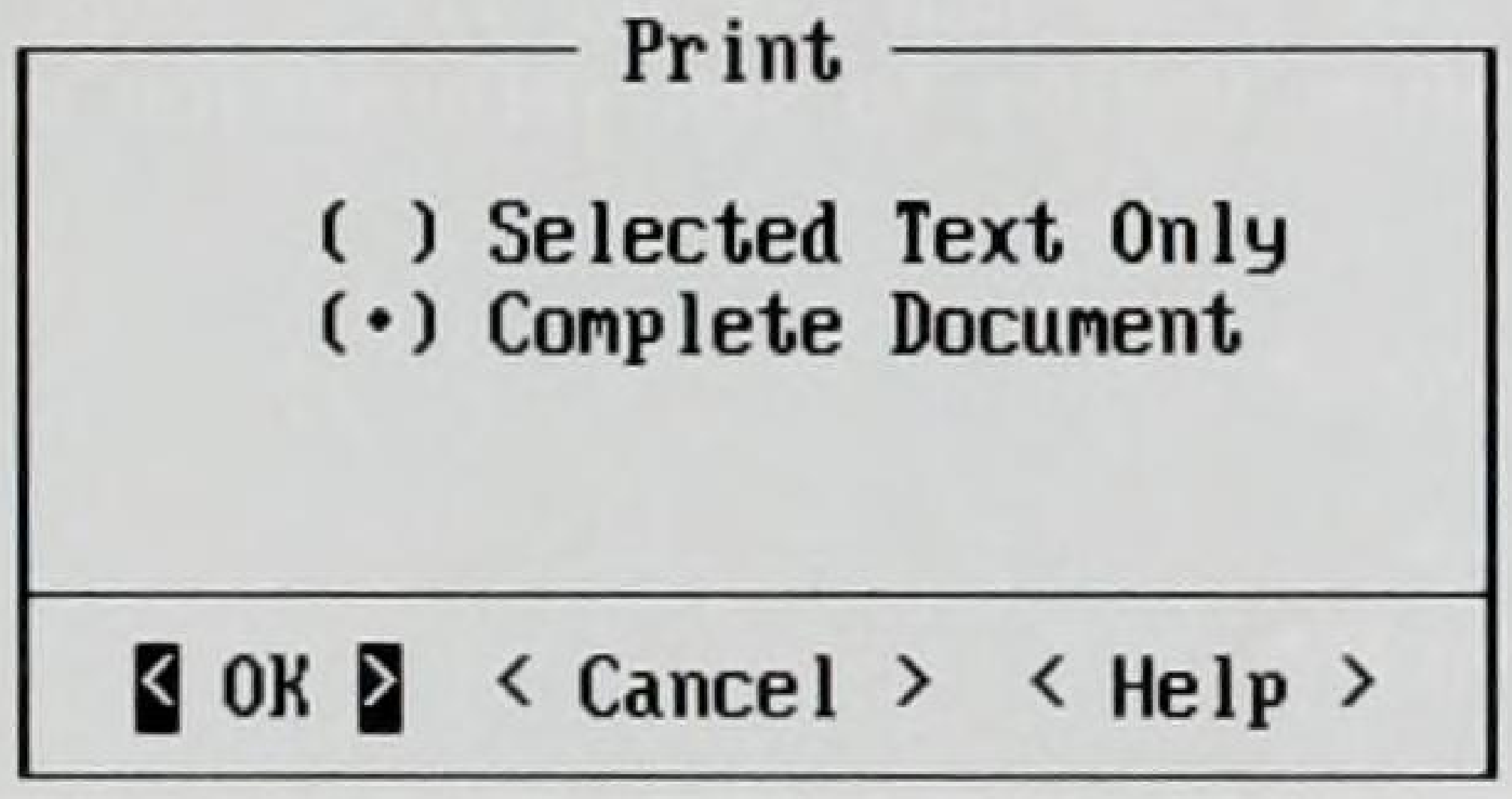
Loaded file is not saved. Save it now?

A *dialog box* is simply a way for a program to ask you for additional information. In some instances dialog boxes will prompt you to type in a file name. In this case, the dialog box wants a Yes or No response. To select a dialog box option, either click on the option with your mouse, or press your keyboard's TAB key to highlight the option desired and then press ENTER. Depending on the dialog box's prompts, there may be times when you can select an option simply by typing the first letter of the option. In most cases, you can cancel a dialog box by pressing ESC.

If you select Yes to save the file's contents, EDIT will save your changes. If you select No instead, EDIT will discard all changes you have made since the last time you saved the file. In either case, EDIT will end, returning control to DOS.

Printing Your File

As you edit a large file, it is often convenient to print all or part of the file's contents without leaving the editor. EDIT's Print option, selected from the File menu, lets you do just that. When you select the Print option, EDIT will display this dialog box:



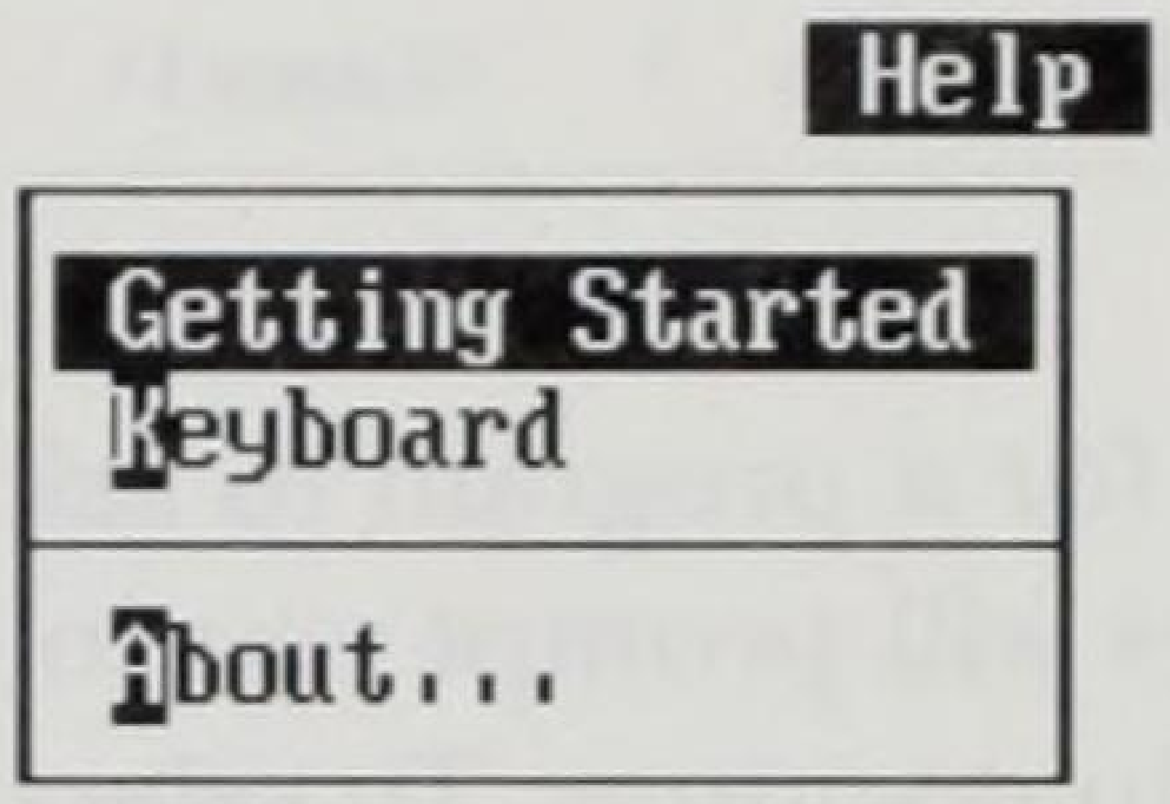
Print

( ) Selected Text Only
(•) Complete Document

The Print dialog box lets you print part of a file (selected text) or the entire document. The "Advanced Concepts" section discusses how to select a specific part of the document. As before, use TAB to select OK (to begin printing) or Cancel (to return to the editor without printing).

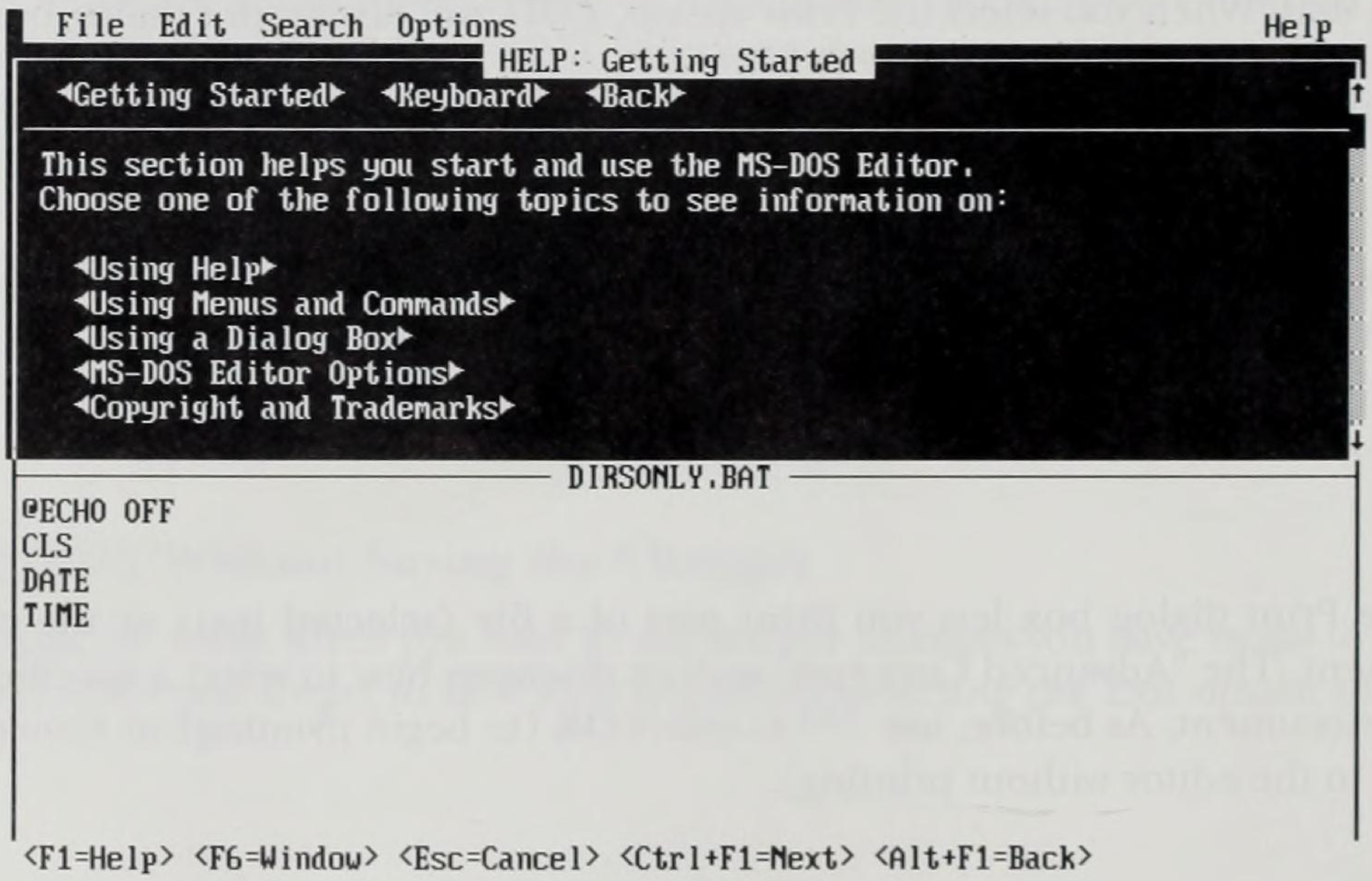
Getting Online Help

Although EDIT does not provide the text-formatting capabilities found in a word processor, EDIT has several powerful features. To help you use these features, EDIT provides built-in online help. There are two ways to invoke Help. First, you can press ALT-H. EDIT will display its Help menu.



The Getting Started option provides information on using Help, menus, dialog boxes, and so on. This option splits your editing window in two, showing the help text in the top window and your file text in the bottom window, as in Figure 37-3.

To display help on a specific option, either press TAB to highlight the option and then press ENTER, or click on the option with your mouse. The Help menu's Keyboard option, for example, displays a window of help on keyboard-related topics, such as the keys you can use to scroll through your file, to select text, or to invoke menus.


File Edit Search Options Help
HELP: Getting Started
◄Getting Started► ◄Keyboard► ◄Back►

```
This section helps you start and use the MS-DOS Editor.
Choose one of the following topics to see information on:

  ◄Using Help►
  ◄Using Menus and Commands►
  ◄Using a Dialog Box►
  ◄MS-DOS Editor Options►
  ◄Copyright and Trademarks►
```

```
@ECHO OFF
CLS
DATE
TIME
```

Figure 37-3. *EDIT's online help divides your screen into two windows, one containing your file, and the other containing help text*

The second way to invoke Help is with the function keys. Note that the bottom line of your screen contains several key definitions. In general, you can press F1 at any time to get help on EDIT or help on an active dialog box. F6 lets you toggle between the Help window and the window containing your file. The ESC key cancels or ends the Help session, which removes the Help window from your screen. Pressing CTRL-F1 displays help on the next topic, letting you quickly traverse through several pages of Help text. If you need to redisplay the previous Help screen, pressing ALT-F1 redisplays Help text in the opposite order of your traversal. That is, pressing ALT-F1 lets you back up through the previous 20 topics.

Use CTRL-F1 and ALT-F1 to view several pages of Help text. Remember, you can end the Help session at any time by pressing ESC.

Table 37-1 briefly describes keys you can use within EDIT's online help.

Getting Around Your Document

When your files are small, the easiest way to move from one location to another in a file is to use your keyboard's arrow and page keys. As your files become larger, however, EDIT provides faster ways to traverse them. Note the vertical and horizontal bars to the right of and below your editing window. These bars are *scroll bars.* As you use the arrow and page keys to move throughout your file, EDIT will move the dark markers along the bars to indicate your relative position in the file. When you are at line 1, column 1, the markers will appear at the top of the vertical scroll bar and at the left of the horizontal scroll bar. As you move down through your text, the dark marker will move down the vertical scroll bar. When you reach the bottom of your file, the marker will be at the bottom of the scroll bar. Likewise, as you move the cursor from left to right across columns that don't currently appear on your screen, EDIT will move the marker along the horizontal scroll bar. By observing the positions of the markers, you can easily determine your position within the file.

Keys and Combinations	**Function**
TAB	Advances to next Help topic
SHIFT-TAB	Moves back to previous Help topic
Character	Moves to next topic beginning with *Character*
SHIFT-*Character*	Moves to previous topic beginning with *Character*
ALT-F1	Displays Help window for previously displayed topic
CTRL-F1	Displays Help window for next topic
SHIFT-CTRL-F1	Displays Help window for previous topic
F6	Toggles between Help window and editing window
ESC	Ends the online Help session

Table 37-1. *EDIT's Online Help Keys and Combinations*

If you are using a mouse, EDIT lets you quickly traverse the file by clicking on a scroll bar marker and *dragging* the marker to a new position on the scroll bar. To drag the marker, position the mouse pointer on the marker and hold the mouse button down while you pull the pointer to the new location; then release the mouse button. If you drag the marker slowly, you can see the text scrolling. If you drag the marker quickly, EDIT will scroll the text past you too fast for you to read.

If you are not using a mouse, Table 37-2 describes keyboard combinations you can use to traverse your file.

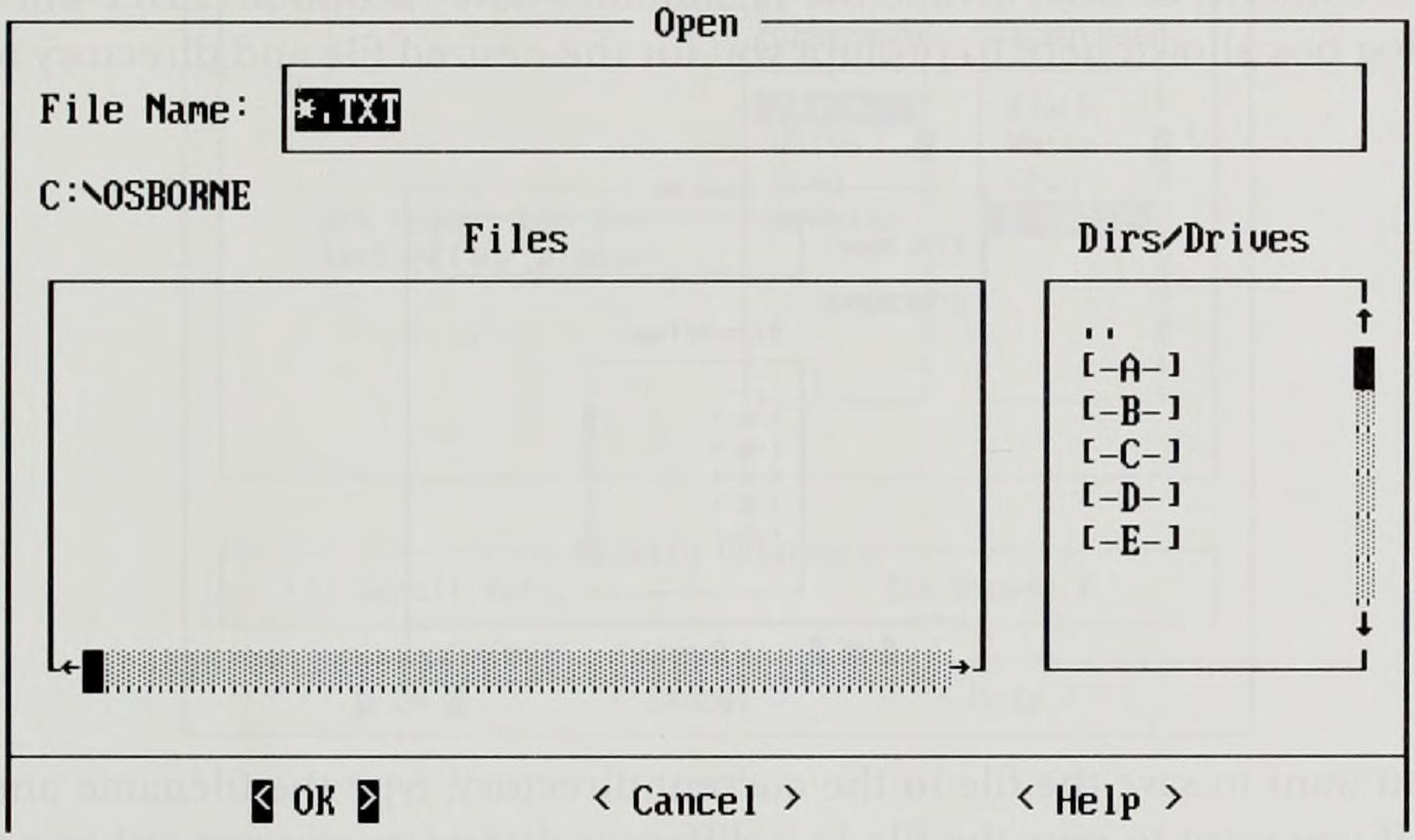
Selecting an Existing File for Editing

The easiest way to edit a file is to invoke EDIT, specifying the filename in EDIT's command line. If you were finished editing one file and you wanted to edit another, it would be inconvenient to exit EDIT just so you could invoke EDIT using a different filename. To let you edit different files, the File menu's Open option lets you either type in the name of the file you want to edit, or select the file from a list of filenames. When you invoke the Open option, EDIT displays the dialog box that is shown in Figure 37-4.

By default, EDIT displays only those files in the current directory with the TXT extension. If you know the name of the file you desire, you can type it in, or you can

Keys and Combinations	**Function**
Arrow keys	Move cursor one position
PGUP	Moves cursor up one text screen
PGDN	Moves cursor down one text screen
HOME	Moves cursor to beginning of line
END	Moves cursor to end of line
CTRL-LEFT ARROW	Moves cursor one word to the left
CTRL-RIGHT ARROW	Moves cursor one word to the right
CTRL-ENTER	Moves cursor to beginning of next line
CTRL-Q-E	Moves cursor to top line of window
CTRL-Q-X	Moves cursor to bottom line of window
CTRL-W or CTRL-UP ARROW	Scrolls text up one line
CTRL-Z or CTRL-DOWN ARROW	Scrolls text down one line
CTRL-HOME or CTRL-Q-R	Moves cursor to beginning of file
CTRL-END or CTRL-Q-C	Moves cursor to end of file
CTRL-PGUP	Scrolls text left one screen
CTRL-PGDN	Scrolls text right one screen

Table 37-2. EDIT's Cursor-Movement and Text-Scrolling Keys and Combinations



Open

File Name: *.TXT

C:\OSBORNE

Figure 37-4. *EDIT's File Open dialog box*

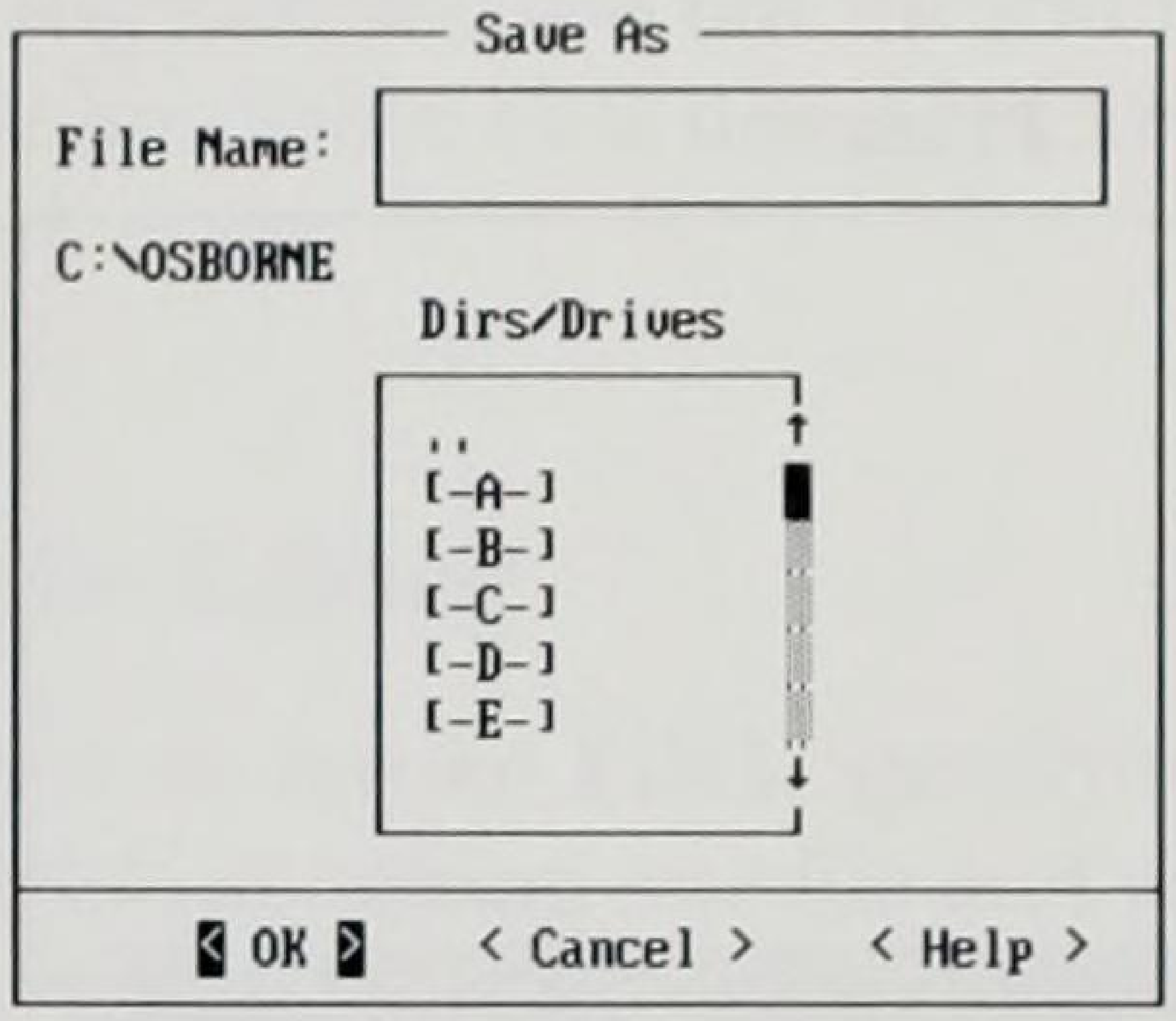
press TAB and your arrow keys to highlight and select the file from the list of names, provided the file is there. If you want to display a different set of files, your batch files, for example, you can change *.TXT to *.BAT and press ENTER. Note the list of directory names that appears in the third box. By pressing TAB to select this box, you can use your arrow keys to highlight the desired directory and press ENTER to select the directory, displaying the list of files matching your filename specification in the file box.

After you select a file, press TAB to highlight the OK option and press ENTER. If you have made changes to your previous file and have not yet saved the changes, EDIT will display a dialog box asking you whether you want to save or discard the changes. If EDIT successfully opens the file you specify, it will display the file's contents in your editing window. If EDIT cannot open or locate the file, it will display a dialog box describing the error it has encountered.

Creating and Saving a New File

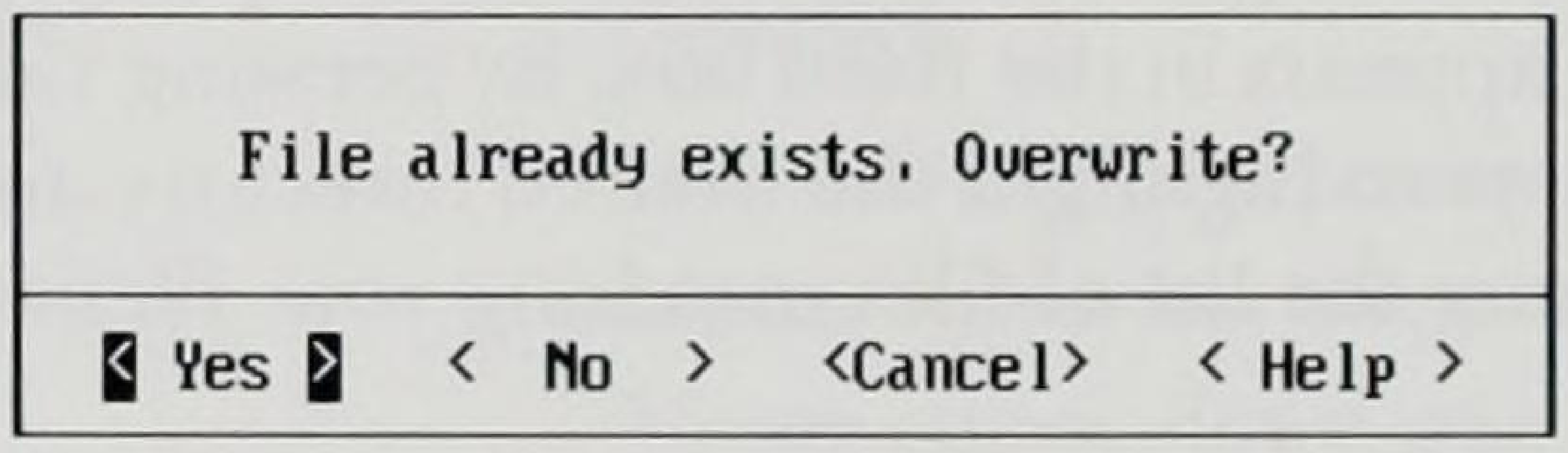
As you just read, EDIT's File menu's Open option lets you open an existing file for editing without having to end and restart EDIT. In a similar way, the File menu's New option lets you create a new file. When you select New from the File menu, EDIT will display an empty editing window to hold your new file. If you have been editing a different file and have not yet saved the file's changes, EDIT will first display a dialog box asking you whether you want to save or discard the changes.

When you select the New option to create a new file, EDIT displays the name Untitled at the top of your window until you save the file's contents to disk. To save the file's contents to disk, invoke the File menu's Save As option. EDIT will display the dialog box shown here to prompt you for the desired file and directory name.



If you want to save the file in the current directory, type the filename and press ENTER. If you want to save the file in a different directory, you can either type in a complete path name or press TAB to select the directory list, then use your arrow keys to highlight the desired directory.

If the filename you specify matches an existing file, EDIT will display a dialog box informing you that a file with the name given exists and asking you if you want to overwrite the file's contents:

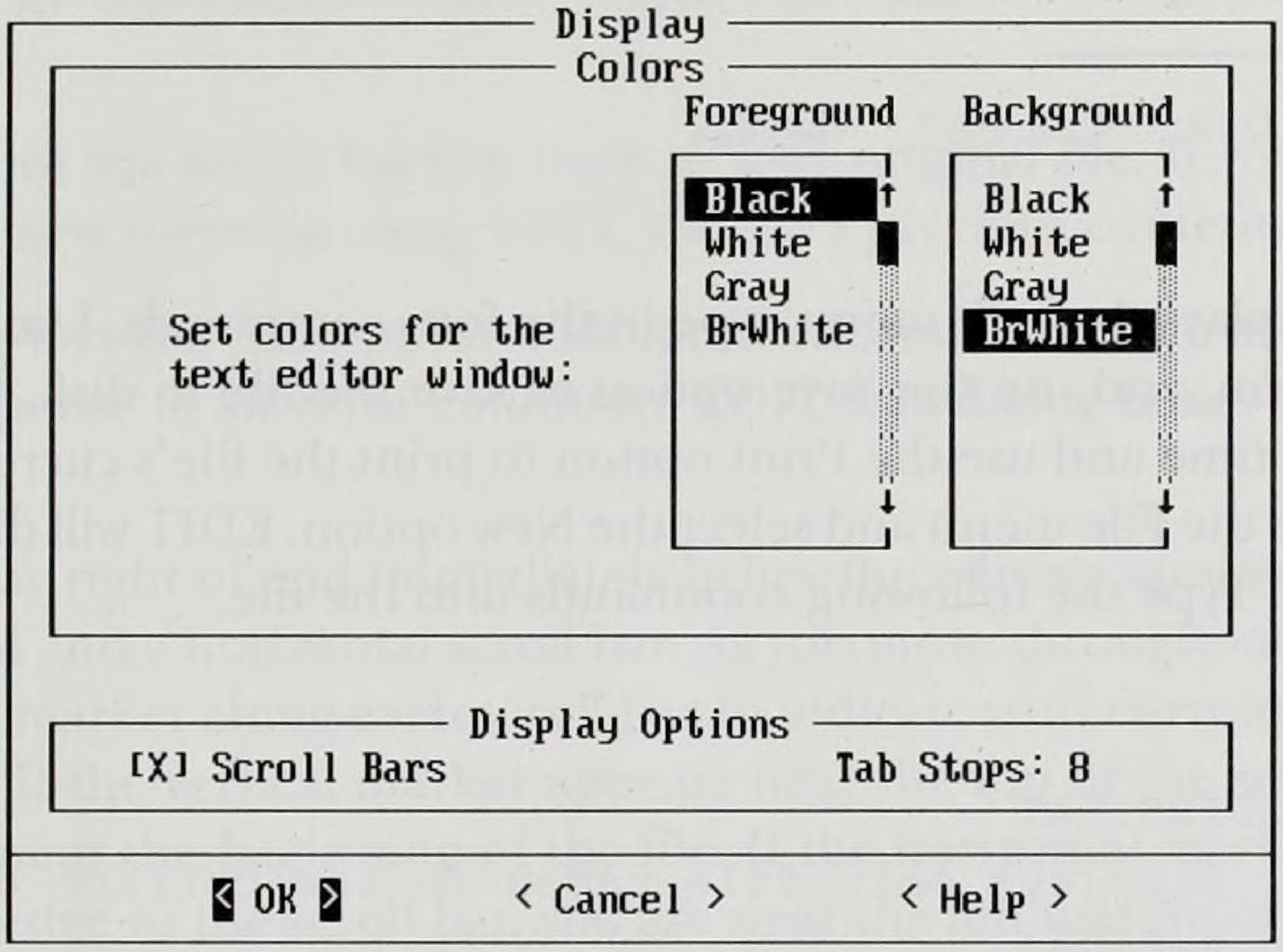

File already exists. Overwrite?

NOTE *Unlike EDLIN and most word processors, EDIT does not create a backup file containing the file's previous contents when you save the file's new changes. If you overwrite an existing file, your file's previous contents are lost.*

Customizing EDIT's Screen Display

The Option menu's Display option lets you customize the foreground and background colors EDIT uses for your editing window, as well as letting you turn off the vertical and horizontal scroll bars to display an extra line and column of your file. When you select Display, EDIT displays the dialog box shown in Figure 37-5.

You can press TAB to toggle between the foreground and background color lists, and then use your keyboard's arrow keys to select different colors. EDIT displays a small box inside the dialog box that illustrates how text will appear in the editing window, based on your current color selections. When you are satisfied with your


Display
Colors
Foreground
Background
Black
White
Gray
BrWhite
Black
White
Gray
BrWhite
Set colors for the text editor window:
Display Options
OK
Cancel
Help

[X] Scroll Bars

Tab Stops: 8

Figure 37-5. Display options dialog box

color selection, select the dialog box's OK option. To leave the existing colors unchanged and cancel the dialog box, press ESC.

The Scroll Bars entry works as a toggle. If you highlight the entry and press SPACEBAR, EDIT will remove the X, telling you scroll bars will not be displayed. If you press SPACEBAR a second time, EDIT will redisplay the X, enabling scroll bars. The Tab Stops option lets you specify the column distances between tab settings. To change the default setting from 8, select the option and type in the new value desired.

Hands On

Using EDIT, create the following batch file DIRSONLY.BAT that displays only the names of directories within the current directory.

```
@ECHO OFF
REM Name: DIRSONLY.BAT
REM Function: Displays directories within the current directory
DIR | FIND  "<DIR>"
```

To begin, invoke EDIT as follows:

```
C:\> EDIT  DIRSONLY.BAT  <ENTER>
```

When EDIT displays the Edit menu, type in the four commands. Use ALT-F to invoke EDIT's File menu, and use the Save option to save the file to disk. Invoke the File menu a second time and use the Print option to print the file's current contents.

Next, invoke the File menu and select the New option. EDIT will display an empty editing window. Type the following commands into the file.

```
@ECHO OFF
REM Name:  NODIRS.BAT
REM Function: Displays only file names in the current directory
DIR | FIND /V "<DIR>"
```

Invoke EDIT's File menu and select the Save As option. When EDIT displays the dialog box prompt for a filename, type in NODIRS.BAT, and press ENTER. Use the File menu's Exit option to end EDIT and return to DOS.

Review

1. How does EDIT differ from EDLIN?
2. What name does EDIT assign to the backup copy of your file's previous contents when you save new changes?
3. What is a dialog box?
4. What are scroll bars?
5. How do you invoke EDIT's online help?

Answers

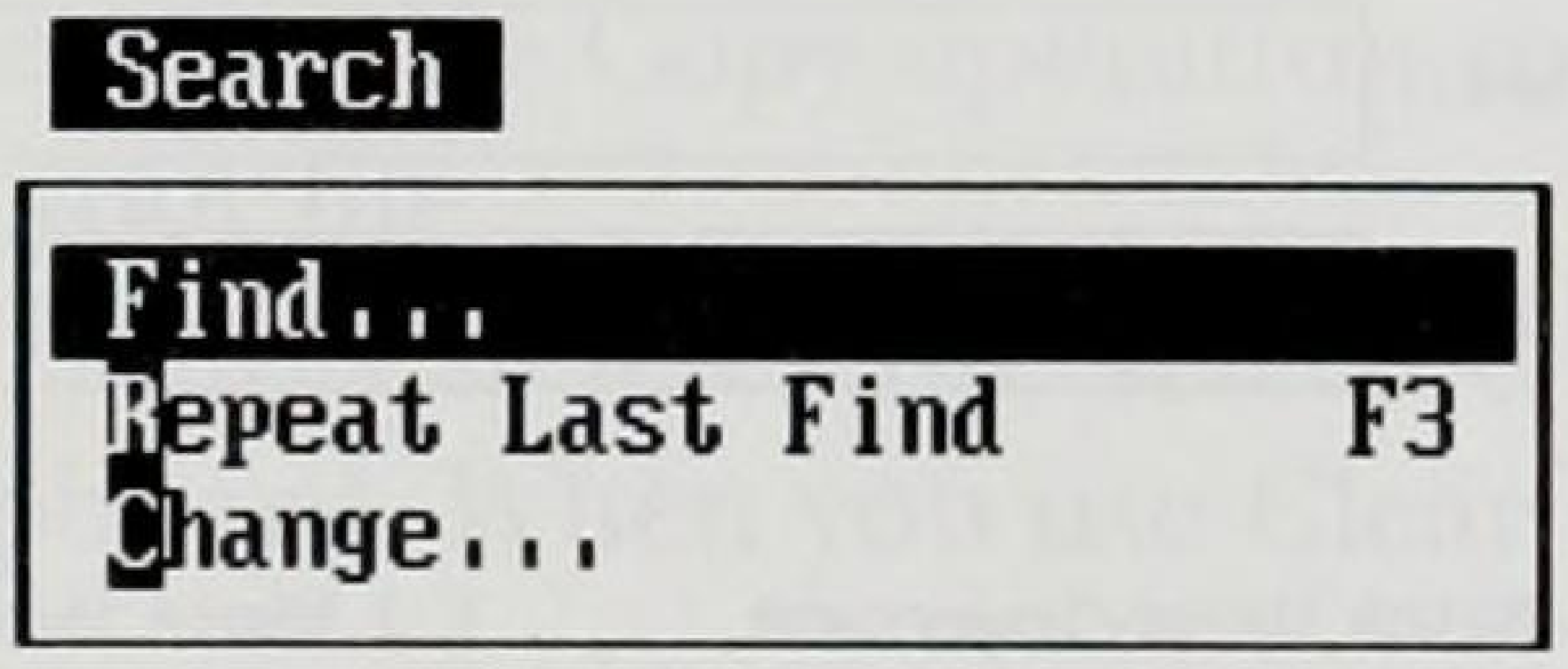
1. EDIT is a full-screen editor that lets you view the file's contents on the entire screen, using a mouse or the arrow and page keys to traverse the file's contents.

EDLIN, on the other hand, only lets you work with one line of the file at a time. In general, EDIT behaves much more like a word processor than does EDLIN.

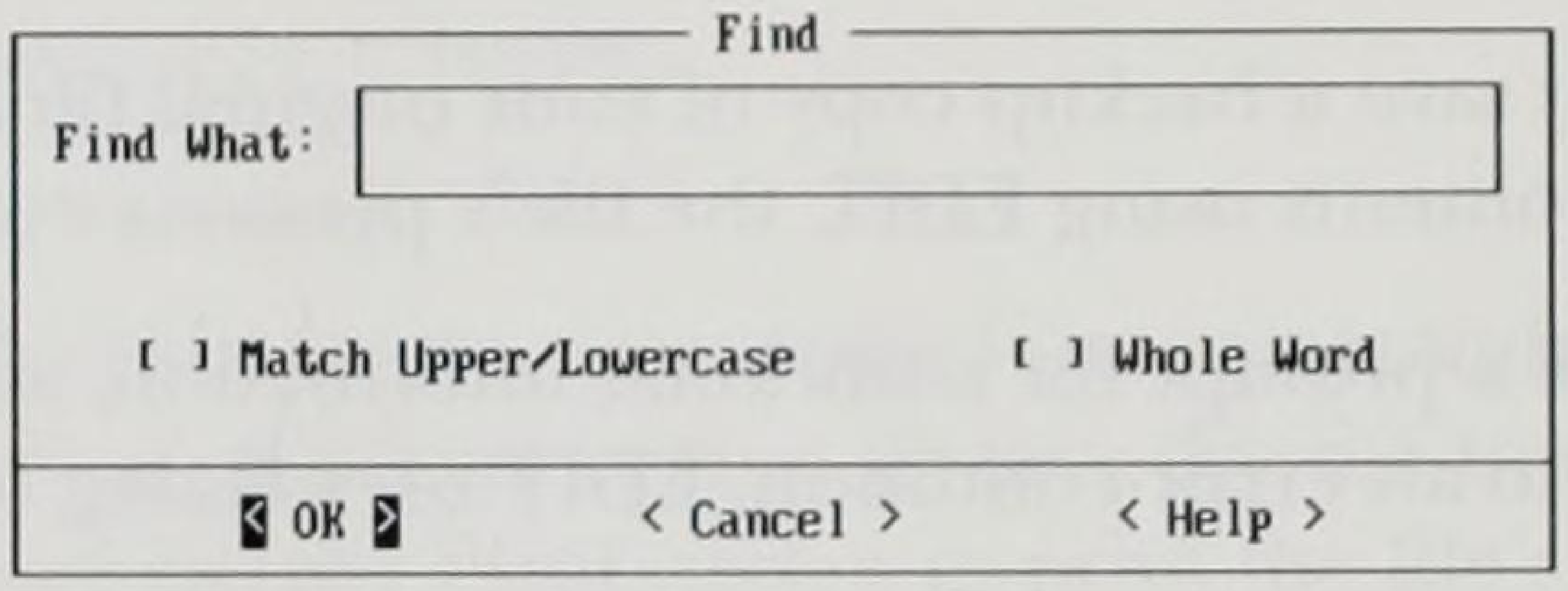
2. EDIT does not save a backup copy of your original file. If you edit a file and save the new contents using EDIT, the file's previous contents are lost.
3. A dialog box is a prompt for additional information, such as a filename or a user response to an error condition. EDIT uses dialog boxes to converse with the user.
4. To the far right of and immediately below the editing window, EDIT displays a vertical and a horizontal scroll bar. As you move throughout your file, EDIT moves a marker along each scroll bar to indicate your current position within the file. If the vertical marker appears near the top of the scroll bar, you are editing near the beginning of the file. If the horizontal marker appears near the left edge of the scroll bar, you are near the left margin of your file. If you are editing the file using a mouse, you can drag the marker to a new position along the scroll bar to select a new position within the file.
5. You can invoke EDIT's online help using ALT-H to select EDIT's Help menu, or you can press F1 at any time to get help on a menu or dialog box, or to access EDIT's general Help facility.

Advanced Concepts

Using the key combinations in Table 37-2, or EDIT's scroll bars and your mouse, you can quickly traverse your files. In addition, EDIT's Search menu lets you specify a word or phrase you want to locate within your document. If EDIT successfully locates your selected text, it will place the cursor at the start of the text. In addition, the Search menu provides a Change option that lets you replace one word or phrase with another. If you invoke the Search menu using ALT-S, EDIT will display the following pull-down menu:



The Find option lets you specify a word or phrase to locate within your document. When you select this option, EDIT displays the Find Text dialog box, shown here:



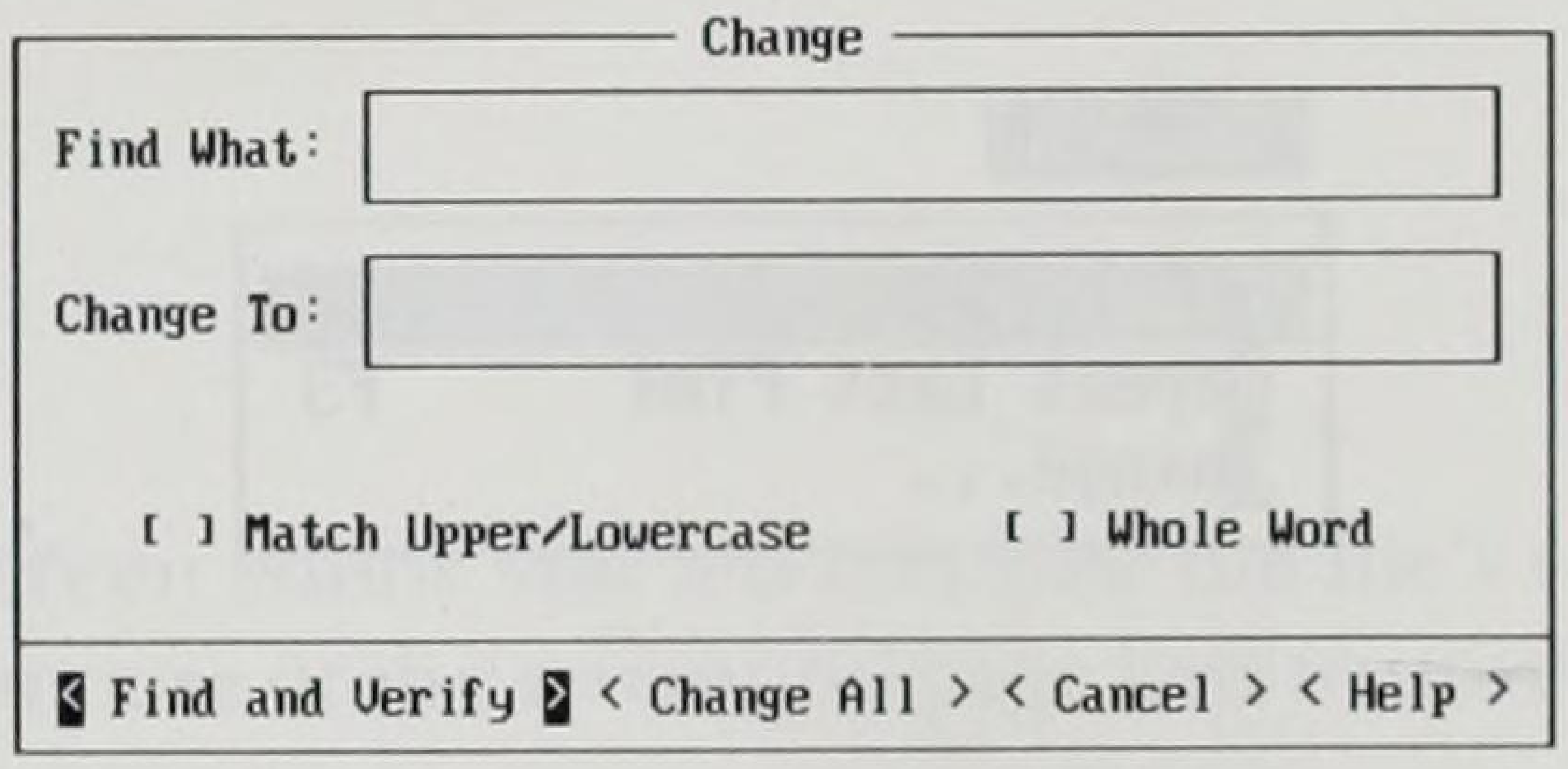
To search a file for the word "Rebels!", for example, you would type **Rebels!** and press ENTER. If EDIT finds matching text, EDIT will advance the cursor to the first match. If the word does not exist in the file, EDIT will display a dialog box stating that it did not find a match.

By default, EDIT considers uppercase and lowercase letters the same, for example, "REBELS!" and "Rebels!". If you want EDIT to perform a case-sensitive search, highlight the Match Upper/Lowercase option and press SPACEBAR. This option works as a toggle. When an X is present within the brackets adjacent to the option, EDIT performs case-sensitive matches. When the X is not present, EDIT considers uppercase and lowercase letters the same.

The Whole Word option is also a toggle. When no X is present, EDIT matches any occurrence of the word, even if the word forms part of a longer word. For example, EDIT would consider "Rebel" and "Rebels!" the same. If you select the Whole Word option, EDIT only matches occurrences of the word that are delimited by blanks, tabs, or other punctuation.

In many situations the text may appear several times throughout your file. The Search menu's Repeat Last Find option lets you quickly repeat the previous search without forcing you to type in the search string a second or third time. Note the "F3" that appears to the right of the option. F3 is a hot key for this option. You can press F3 at any time to repeat the last search operation without having to invoke the Search menu and select this option. Hot keys save you time and keystrokes.

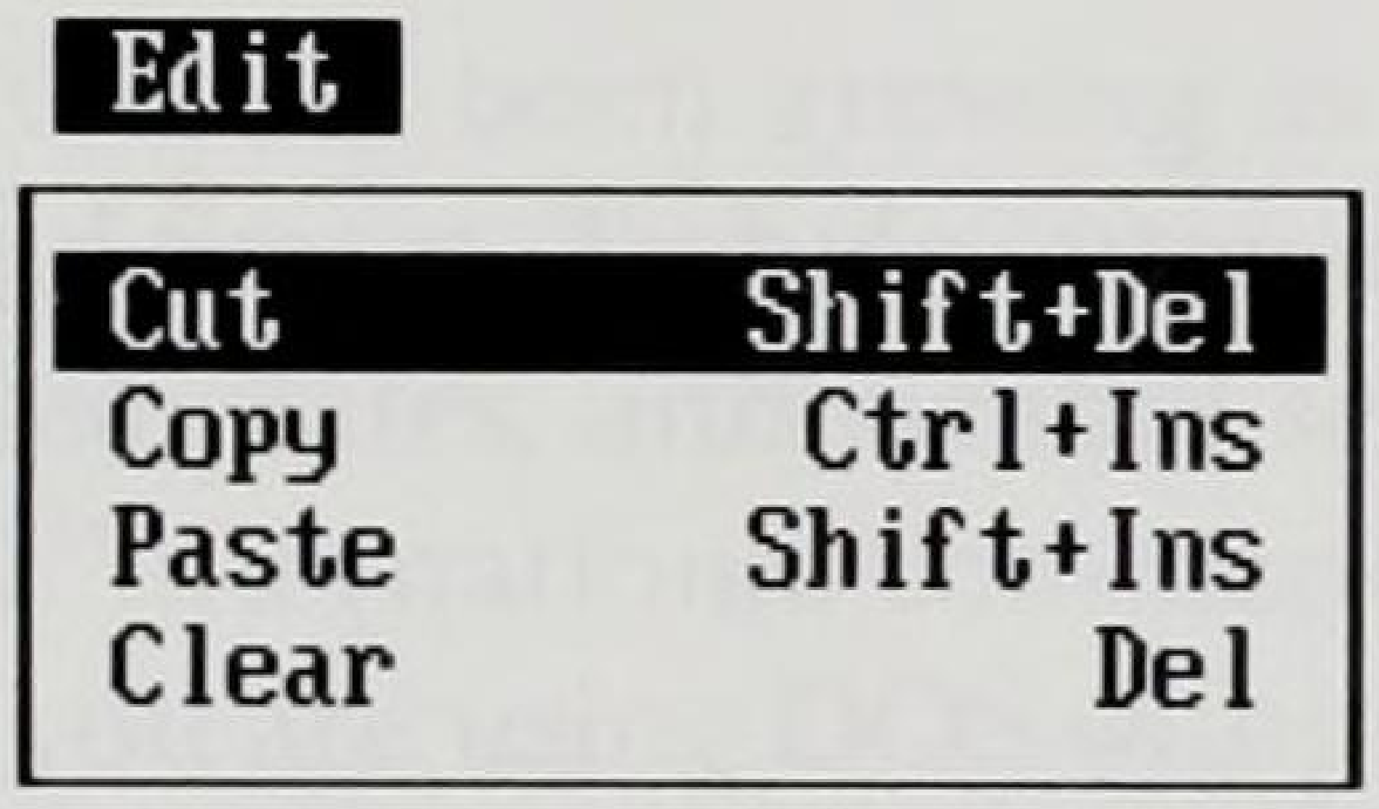
The Search menu's Change option lets you replace one word or phrase with another. When you select this option, EDIT will display the dialog box shown here:



The Find What option prompts you for the word or phrase you want to change. Type in the word or phrase and press TAB. The Change To entry is the new word or phrase you want to use. Again, type in the word or phrase and press TAB. The Find and Verify option directs EDIT to locate each matching word or phrase, and then to display a dialog box to determine whether you want to replace the match. The Change All option directs EDIT to automatically change each match without asking you for verification. As with all dialog boxes, you can cancel the Text Change dialog box at any time by pressing ESC.

Performing Cut-and-Paste Operations

Cutting and pasting is the process of moving text from one location in your file to another. To begin, you must select the lines of text you want to cut and paste. To select text, hold down a SHIFT key and press UP ARROW or DOWN ARROW (or PGUP and PGDN for large amounts or text). EDIT will display the selected text in reverse video. To select text using your mouse, move the mouse pointer to the beginning of the text. Next, hold down the left mouse button while you move the mouse. As before, EDIT will display the selected text in reverse video. After you select the text, you can invoke EDIT's Edit menu using ALT-E. EDIT will display the following menu options:



Note the available hot keys for each menu option. The Cut option removes the selected text from your file and stores the text in a location you can't view, called the clipboard. You use the Cut option when you want to move text from one location to another. The Cut option's counterpart is the Paste option, which copies the clipboard's current contents to the current cursor location in your file. To cut and paste text, you select the text to move, cut the text from the file to the clipboard, move the cursor to the location in the file where you want the text, and then paste the clipboard's contents into the file.

The Copy option copies selected text from the file to the clipboard without removing the text from the file. The Copy operation combines with Paste to let you duplicate text throughout your file.

Last, the Clear option simply deletes the selected text from your file without copying the text to the clipboard. When you use Clear to remove selected text, you can't get the text back.

EDIT does not provide a menu option to merge one file's contents into another. However, if you select all the text in one file and copy the text to the clipboard, you can then open a second file and paste the clipboard's contents into that file, merging the two files as desired.

NOTE

Key Terms

Screen editor A text editor that lets you view and edit a file's contents on the entire screen, using your keyboard's arrow and page keys to traverse the file.

Cut and paste The process of moving (cutting) text from one location in the file and placing the text (pasting) at a new location.

Scroll bar A vertical bar at the right side of a window, or a horizontal bar at the bottom of a window, that displays a marker indicating your relative location within the file and that lets you quickly change locations within the file by dragging the marker to a new location on the scroll bar.

Dialog box A box that a program displays to communicate with the user, either to obtain additional information, such as a filename, or to obtain a specific user response to a limited number of options.

Chapter 38

Support for International Users

Over the past few years, DOS use has been growing as rapidly in countries around the world as it has in the United States. To help international users work with their own character sets, keyboard templates, and symbol sets (date, time, and currency formats), DOS provides several "international commands." By default, DOS uses the United States configuration. If you are using DOS within the United States and don't need to correspond with international users using another country's character set, you can skip ahead to Chapter 39. If you are an international DOS user, you can finally learn how to customize your computer.

Selecting a Code Page

A *character set* is a collection of letters and punctuation symbols—in other words, a set of characters. Just as different countries use different languages, most countries have unique character sets. DOS refers to character sets as *code pages.* To let international users work with their own character sets, DOS provides support for many different code pages.

Code pages are similar in concept to typewriters that let you select a new set of letters by changing the typewriter ball as shown in the following illustration.

When you tell DOS to change code pages, DOS, like the typewriter, will use the new character set. DOS provides support for six different code pages as listed in Table 38-1.

Appendix C presents all the characters in each code page. Take a moment to examine the code pages to determine which contains the character set you need. Most of the commands presented in this chapter require you to specify a code-page value, as shown in Table 38-1.

Setting an International Configuration

To set up an international system, you have to perform several steps. Fortunately, once you determine the commands you need, you can place the commands and device-driver entries in your AUTOEXEC.BAT and CONFIG.SYS files and forget about them. Briefly, to set up an international system you must perform the following steps:

1. Select the country's symbol set using the CONFIG.SYS COUNTRY entry.
2. Install video code-page support with the DISPLAY.SYS device driver.
3. Install printer code-page support with the PRINTER.SYS device driver.
4. Optionally, invoke the NLSFUNC command to define country-specific symbols and enable code-page switching.
5. Select the country's keyboard template using the KEYB command.
6. Load code-page support for the keyboard, screen, and printer using the MODE command.

Country	Code-Page Value
United States	437
Multilingual	850
Slavic	852 (DOS 5 or later)
Portuguese	860
French Canadian	863
Nordic	865

Table 38-1. *Code Pages Supported by DOS*

Loading Key Device Drivers

For DOS versions 3.3 and later, the DISPLAY.SYS and PRINTER.SYS device drivers, briefly discussed in Chapter 24, provide code-page support for your video display and printer. The format of the DISPLAY.SYS device driver entry is

```
DEVICE=C:\DOS\DISPLAY.SYS CON[:]=(DisplayType[,[CodePage]
        [,CodePageCount]])
```

or for video devices supporting subfonts:

```
DEVICE=C:\DOS\DISPLAY.SYS CON[:]=(DisplayType[,[CodePage]
        [,(CodePageCount, SubfontCount)]])
```

DisplayType specifies your video display. You must specify one of the following types:

EGA (use for EGA and VGA systems)
LCD (liquid-crystal display for portables)
CGA
MONO

CodePage is one of the three-digit code-page values listed in Table 38-1. *CodePage Count* specifies the number of additional code pages your video system can support, from 0 to 6. For LCD systems the maximum value is 1. For EGA and VGA systems, the maximum value is 6. For CGA and MONO systems, the value is 0. *SubFontCount* specifies the number of font subsets your hardware supports for each code page. For

EGA systems, the default value is 2. For LCD systems, the default is 1. Most users won't require subfonts.

The following DISPLAY.SYS entry loads code page 437 as the primary code page on an EGA (or VGA) system, specifying two additional code pages:

```
DEVICE=C:\DOS\DISPLAY.SYS  CON:=(EGA, 437, 2)
```

In a similar way, the PRINTER.SYS device driver provides code-page support for parallel printers. The format of PRINTER.SYS is

```
DEVICE=C:\DOS\PRINTER.SYS  LPTn:=(PrinterType [,[CodePage]
      [,CodePageCount]])
```

LPT*n* specifies the printer port, LPT1 through LPT3. *PrinterType* specifies one of the printers listed in Table 38-2.

As you can see, PRINTER.SYS restricts its code-page support to a select group of IBM printers. If your printer is not listed, your printer may be compatible with one of the three. Check your printer manual for details on code-page support. *CodePage* and *CodePageCount*, as before, specify the primary code page and the maximum number of additional code pages the printer supports.

The following PRINTER.SYS entry installs code-page support for an IBM 4208 Proprinter:

```
DEVICE=C:\DOS\PRINTER.SYS  LPT1:=(4208, 437, 2)
```

Selecting a Symbol Set

Most countries have a unique date and time format, as well as unique currency formats. As briefly discussed in Chapter 24, the CONFIG.SYS COUNTRY entry lets you select an international symbol set. The format of the COUNTRY entry is

```
COUNTRY=CountryCode[,[CodePage][,CountryInfoFile]]
```

CountryCode is a three-digit number that identifies the desired country. Table 38-3 defines the country-code values, as well as the code-page values normally used for the country.

Printer Type	Printer
4201	IBM 4201 or 4202 Proprinter
4208	IBM 4207 or 4208 Proprinter
5202	IBM 5202 Quietwriter III

Table 38-2. Printer Types Supported by PRINTER.SYS

Country	Country Code	Code Page
United States	001	437 or 850
French Canadian	002	863 or 850
Latin America	003	850 or 437
Netherlands	031	850 or 437
Belgium	032	850 or 437
France	033	850 or 437
Spain	034	850 or 437
Hungary	036	852 or 850
Yugoslavia	038	852 or 850
Italty	039	850 or 437
Switzerland	041	850 or 437
Czechoslovakia	042	852 or 850
United Kingdom	044	437 or 850
Denmark	045	850 or 865
Sweden	046	850 or 437
Norway	047	850 or 865
Poland	048	852 or 850
Germany	049	850 or 437
Brazil	055	850 or 437
English (International)	061	437 or 850
Australia	061	437 or 850
Japan	081*	932* or 437
Korea	082*	934* or 437
Chinese (simplified)	086*	936* or 437
Chinese (traditional)	088*	938* or 437
Portugal	351	850 or 860
Finland	358	850 or 437
Arabic countries	785*	864* or 850
Israel	972*	862* or 850

Table 38-3. Country-Code Values and Code-Page Combinations (indicates a symbol set or code page only available with special DOS versions)*

CountryInfoFile specifies the path name of the file containing the international symbol sets. Most users will use the DOS file COUNTRY.SYS.

To select the symbol set for Germany, for example, you would use the following CONFIG.SYS COUNTRY entry:

```
COUNTRY=049,850,C:\DOS\COUNTRY.SYS
```

Likewise, to select Finland you would use this entry:

```
COUNTRY=358,850,C:\DOS\COUNTRY.SYS
```

DOS 3.2 and Earlier

If you are using DOS 3.2 or earlier, only specify the country code in the COUNTRY entry. DOS did not use code pages or the COUNTRY.SYS file until version 3.3. In DOS 3.2, the following COUNTRY entry selects the German symbol set:

```
COUNTRY=049
```

Selecting a Keyboard Template

Just as each country has its own characters and symbol sets, most countries use a unique keyboard layout (called the *keyboard template*). Appendix C displays all the keyboard templates. Beginning with DOS 3.3, the KEYB command loads memory-resident software that lets you select a keyboard layout. The format of KEYB is as follows:

```
KEYB KeyboardCode[,[CodePage][,KeyboardFile]][/E][/ID:nnn]
```

KeyboardCode is a unique two-letter code that identifies the desired keyboard template. Table 38-4 identifies the keyboard codes, code pages, and secondary identification values for cases where the country has two templates.

KeyboardFile is the complete path name to the file containing the keyboard templates, which is typically the file KEYBOARD.SYS. The /E switch informs DOS you are using an enhanced keyboard. The /ID switch lets you select one of two possible templates by specifying a secondary identification value.

The following KEYB command selects the German keyboard template:

```
C:\> KEYB  GR,850,C:\DOS\KEYBOARD.Sys <ENTER>
```

KEYB does not force you to include a code-page value. If the code page has not yet been defined and you include the code-page value, the command will fail. The

Country	Keyboard Code	Code Page	Secondary Identification
United States	US	437 or 850	103
French Canadian	CF	863 or 860	058
Latin America	LA	850 or 437	171
Netherlands	NL	850 or 437	143
Belgium	BE	850 or 437	120
France	FR	437 or 850	120, 189
Spain	SP	850 or 437	172
Italy	IT	850 or 437	141, 142
Switzerland German	SG	850 or 437	000
Switzerland French	SF	850 or 437	150
United Kingdom	UK	437 or 850	166, 168
Denmark	DK	850 or 865	159
Sweden	SV	850 or 437	153
Norway	NO	850 or 865	155
Germany	GR	850 or 437	129
Brazil	BR	437 or 850	
Portugal	PO	850 or 860	163
Finland	SU	850 or 437	153
Czechoslovakia	CZ	852 or 850	
Slavic	SL	852 or 850	
Hungary	HU	852 or 850	
Poland	PL	852 or 850	
Yugoslavia	YU	852 or 850	

Table 38-4. *International Keyboard Templates*

following KEYB command selects the German keyboard template without specifying the code-page value:

```
C:\> KEYB  GR,,C:\DOS\KEYBOARD.SYS <ENTER>
```

If you are using DOS 4, use the CONFIG.SYS INSTALL entry to invoke KEYB as shown here:

```
INSTALL=C:\DOS\KEYB.COM  GR,850,C:\DOS\KEYBOARD.SYS
```

If you are using DOS 5 or later and upper memory, use the LOADHIGH command to install KEYB.

```
C:\> LOADHIGH C:\DOS\KEYB.COM  GR,850,C:\DOS\KEYBOARD.SYS <ENTER>
```

If you invoke KEYB without specifying any parameters, KEYB will display information about the current keyboard template.

```
Current keyboard code: GR  code page: 850
Current CON code page: 850
```

If you have not yet prepared a code page for use using MODE, KEYB will display the following message:

```
KEYB has not been installed
Active code page not available from CON device
```

Once you select a keyboard template you can switch between the default and current template using the CTRL-ALT-F1 and CTRL-ALT-F2 keyboard combinations. To select the default template press CTRL-ALT-F1. To select the new template press CTRL-ALT-F2.

DOS 3.2 and Earlier

KEYB was introduced with DOS 3.3. Before version 3.3, DOS provided individual commands for each keyboard template, as listed in Table 38-5.

Command	Country
KEYBFR	French
KEYBGR	German
KEYBIT	Italian
KEYBSP	Spanish

Table 38-5. Keyboard Template Commands Prior to DOS 3.3

Preparing Code Pages for Use

In Chapter 13 you used MODE to redirect printer output of a parallel printer to a serial port. In Chapter 41 you will use the MODE command to customize your serial ports, screen, and keyboard. In this section you will use MODE to prepare and select device code pages.

The MODE command lets you select a code page for use or prepare code pages for future use. To provide code-page support, DOS includes five files with the CPI (code-page information) extension, as described in Table 38-6.

Preparing Device Code Pages

Before you can select a code page for use, you must prepare the device. In DOS 4 or later, the MODE command lets you prepare (download) code-page information for a device as follows:

```
MODE DeviceName  CODEPAGE  PREPARE=((CodePages) CodePageFile)
```

In DOS 3.3 you only need one set of parentheses.

```
MODE DeviceName CODEPAGE  PREPARE=(CodePages) CodePageFile
```

DeviceName specifies the device you want to prepare for code-page use, either CON, LPI1, LPT2, or LPT3. The entries CODEPAGE and PREPARE tell MODE the operation you desire. MODE lets you abbreviate CODEPAGE as CP and PREPARE as PREP. *CodePages* tells MODE the code-page values the device may use, as described in Table 38-1. If you want MODE to prepare two or more code pages, separate the code-page values with commas. *CodePageFile* specifies the complete path name to one of the CPI files listed in Table 38-6.

In DOS 4 or later, the following command prepares the console device CON to use code pages 437, 850, and 860:

```
C:\> MODE CON: CP PREP=((437,850,860) C:\DOS\EGA.CPI) <ENTER>
```

Code Page File	**Device Support**
EGA.CPI	EGA
LCD.CPI	Liquid-crystal display (portables)
4201.CPI	IBM 4201 and 4202 Proprinter
EGA.CPI	IBM 4207 and 4208 Proprinter

Table 38-6. DOS Code-Page Support Files

In DOS 3.3 an equivalent command is

```
C:\> MODE CON: CP PREP=(437,850,860) C:\DOS\EGA.CPI <ENTER>
```

In DOS 4 or later, the following MODE command prepares code pages for a 4201 Proprinter:

```
C:\> MODE LPT1: CP  PREP=((437,850)  C:\DOS\4201.CPI) <ENTER>
```

In earlier versions, remove the second set of parentheses.

```
C:\> MODE LPT1: CP  PREP=(437,850)  C:\DOS\4201.CPI <ENTER>
```

If MODE cannot locate the CPI file, MODE will display the following error message:

```
Failure to access code page font file
```

If this error message appears, make sure you are specifying a correct path name to the CPI file.

If MODE successfully prepares the device, MODE will display the following:

```
MODE prepare code page function completed
```

Selecting Device Code Pages

After you prepare code pages for use, you can use the MODE CODEPAGE SELECT command to select a specific code page for use. The format of the command is as follows:

MODE *DeviceName* CODEPAGE SELECT=*CodePage*

As before, *DeviceName* specifies the desired device, and *CodePage* is a three-digit code-page value. The following MODE command selects code page 850 for the CON device.

```
C:\> MODE CON CODEPAGE SELECT=850 <ENTER>
```

MODE lets you abbreviate SELECT as SEL. If MODE is successful, MODE will display the following:

```
MODE select code page function completed
```

If the code page specified has not yet been prepared for use, MODE will display the following error message:

```
Code page not prepared
```

Display Device Code-Page Information

After you prepare and select code pages, you may be interested in the current device configuration. Using the CODEPAGE /STATUS switch, you can display device code-page information as shown here:

```
C:\> MODE  CON  /STATUS <ENTER>

Status for device CON:
----------------------
Columns=80
Lines=25

Active code page for device CON is 437
Hardware code pages:
  code page 437
Prepared code pages:
  code page 437
  code page 850

MODE status code page function completed
```

The MODE command lets you abbreviate /STATUS as /STA.

Refreshing a Code Page

When you prepare a device for code-page use, MODE downloads code-page information into the device's memory. If the device loses power, the code-page information is lost. Using the CODEPAGE REFRESH command, you can reinstate prepared code pages. The following MODE command, for example, refreshes the code pages previously prepared for LPT1:

```
C:\> MODE LPT1: CODEPAGE REFRESH <ENTER>
```

MODE lets you abbreviate REFRESH as simply REF.

Changing Code Pages with CHCP

The DOS MODE command lets you select code pages for devices on an individual basis. Using the CHCP command, you can change code pages for your screen and printer in one step. Before you can use CHCP, you must invoke the NLSFUNC (National Language Support Function) command. NLSFUNC lets you specify the directory containing the country-specific symbol-set file COUNTRY.SYS and enables code-page switching via CHCP.

If you are using DOS 3.3, invoke NLSFUNC as follows:

```
C:\> NLSFUNC  C:\DOS\COUNTRY.SYS <ENTER>
```

If you are using DOS 4, invoke NLSFUNC using the CONFIG.SYS INSTALL entry shown here:

```
INSTALL=C:\DOS\NLSFUNC.EXE  C:\DOS\COUNTRY.SYS
```

Finally, in DOS 5 or later, use either INSTALL or the LOADHIGH command:

```
C:\> LOADHIGH  C:\DOS\NLSFUNC.EXE  C:\DOS\COUNTRY.SYS <ENTER>
```

After you invoke NLSFUNC, you can change code pages using CHCP. The format of CHCP is as follows:

```
CHCP  [CodePage]
```

CodePage is the three-digit value corresponding to a code page (see Table 38-1). If you don't specify a code page, CHCP will display the active code-page setting.

```
C:\> CHCP <ENTER>
Active code page: 437
```

The following command activates code page 850. If you have not yet prepared a device for the code page specified, CHCP will display the following error message:

```
C:\> CHCP  850 <ENTER>
Invalid code page
```

SUMMARY

DISPLAY.SYS

The DISPLAY.SYS device driver provides video code-page support for international users. DISPLAY.SYS lets you specify your video display type and hardware support for code pages. The following entry selects code-page support for an EGA (and VGA system):

```
DEVICE=C:\DOS\DISPLAY.SYS CON:=(EGA,437,2)
```

The value 437 specifies the primary code page, while the value 2 specifies the number of additional code pages supported.

PRINTER.SYS

The PRINTER.SYS device driver provides code-page support for a limited number of IBM printers. If your printer is not listed, check your printer documentation for code-page support. The following PRINTER.SYS entry loads code-page support for an IBM 4201 printer:

```
DEVICE=C:\DOS\PRINTER.SYS  LPT1:=(4201, 437, 1)
```

COUNTRY

The CONFIG.SYS COUNTRY entry lets you select an international symbol set. The symbol set defines the country's date and time formats, currency format, and collating order for sort operations. The following COUNTRY entry selects the French Canadian symbol set:

```
COUNTRY=002,C:\DOS\COUNTRY.SYS
```

KEYB

KEYB lets you select an international keyboard template. Once you select a template, you can switch between the default and new template using the following keyboard combinations.

SUMMARY

Ctrl Alt F1	Default keyboard template
Ctrl Alt F2	International keyboard template

The following KEYB command selects the Finnish keyboard:

```
C:\> KEYB  SU,850,C:\DOS\KEYBOARD.SYS  <ENTER>
```

MODE Code-Page Commands

Before a device can use code pages, you must prepare the device for code-page use. Using code-page font information stored in one of five CPI files, the MODE command lets you download code-page information to a device. The following DOS 5 or later MODE command, for example, prepares the console to use code pages 437 and 850:

```
C:\> MODE  CON: CP PREP=((437,850) C:\DOS\EGA.CPI)  <ENTER>
```

After you prepare a device for code-page use, you can select a specific code page using MODE's CODEPAGE SELECT command.

```
C:\> MODE  CON:  CP  SEL=850  <ENTER>
```

Finally, using the /STATUS switch, you can display a device's code-page information as follows:

```
C:\> MODE  CON:  CP  /STA <ENTER>
```

SUMMARY

NLSFUNC and CHCP

The NLSFUNC command enables code-page switching by CHCP. The format of NLSFUNC is

```
C:\> NLSFUNC  C:\DOS\COUNTRY.SYS <ENTER>
```

When you select a code page using CHCP, DOS automatically selects the code page for all devices supporting code-page switching. To select code page 850 using CHCP, issue the following command:

```
C:\> CHCP  850 <ENTER>
```

Before you can select a code page for use with CHCP, you must first prepare the device for the code page using MODE.

Hands On

This section presents the steps you would follow to configure a system for use in Germany. Following these same steps, substitute your country's code page, country code, and keyboard template values.

First, load the DISPLAY.SYS device driver in CONFIG.SYS. The following entry assumes a VGA or EGA monitor:

```
DEVICE=C:\DOS\DISPLAY.SYS  CON:=(EGA,437,2)
```

Second, place a COUNTRY entry in CONFIG.SYS.

```
COUNTRY=049,437,C:\DOS\COUNTRY.SYS
```

If you are using one of the printers listed in Table 38-2, install the PRINTER.SYS device driver.

Next, invoke KEYB from within your AUTOEXEC.BAT file to select the German keyboard template.

```
KEYB  GR,437,C:\DOS\KEYBOARD.SYS
```

If you are using DOS 4 use the CONFIG.SYS INSTALL entry to load KEYB.

Again, from within AUTOEXEC.BAT or using the CONFIG.SYS install entry, invoke NLSFUNC.

```
NLSFUNC  C:\DOS\COUNTRY.SYS
```

Next, from within your AUTOEXEC.BAT file, invoke MODE to prepare the code page. In DOS 4 or later use the following:

```
MODE  CON:  CODEPAGE  PREP=((437,850) C:\DOS\EGA.CPI)
MODE  CON:  CODEPAGE  SELECT=437
```

In DOS 3.3, use the following:

```
MODE  CON:  CODEPAGE  PREP=(437,850) C:\DOS\EGA.CPI
MODE  CON:  CODEPAGE  SELECT=437
```

Using CTRL-ALT-DEL, restart your system in its international configuration.

Review

1. What is a code page?
2. What is an international symbol set?
3. What is a keyboard template?
4. What are CPI files?
5. How does selecting a code page using CHCP differ from using MODE?

Answers

1. A code page is a character set that contains letters, numbers, and punctuation symbols. DOS provides code-page (character-set) support for international users.
2. An international symbol set specifies a country's date, time, and currency formats, as well as collating sequences for sorting operations. The United States, for example, specifies dates as *MM-DD-YY*. In Germany, however, the date is specified as *DD.MM.YY*.
3. A keyboard template is the mapping of keys on the keyboard to letters, numbers, or symbols in the current character set (code page). To assist international users, DOS provides keyboard templates for each country.

4. CPI stands for code-page information. A file with the CPI extension contains code-page information for the device specified in the base name. For example, the file EGA.CPI contains code-page information for EGA (as well as VGA) monitors.

5. The CHCP command lets you change the code page used by the screen and printer in one step. The MODE command changes the code page for your devices one device at a time.

Advanced Concepts

As you know, the IBM PC lets you use ASCII and extended ASCII characters. Appendix B illustrates the extended ASCII character set. If you examine the code-page values in Appendix C, you will find the international character sets all reside in the range of extended ASCII characters (128 through 255). Depending on the age of your monitor, your monitor may not be able to display these characters clearly in different graphics modes. If your monitor has difficulty displaying such characters, you will need to invoke the GRAFTABL command discussed in Chapter 41. GRAFTABL loads memory-resident software that helps your computer display these characters correctly. DOS 6 does not support the GRAFTABL command.

Key Terms

Code page A character set containing letters, symbols, and numbers.

Code-page switching The process of changing the character set a device uses to display output; remapping the keyboard for use with international symbols.

Chapter 39

Using the DOS 5 and 6 Shell

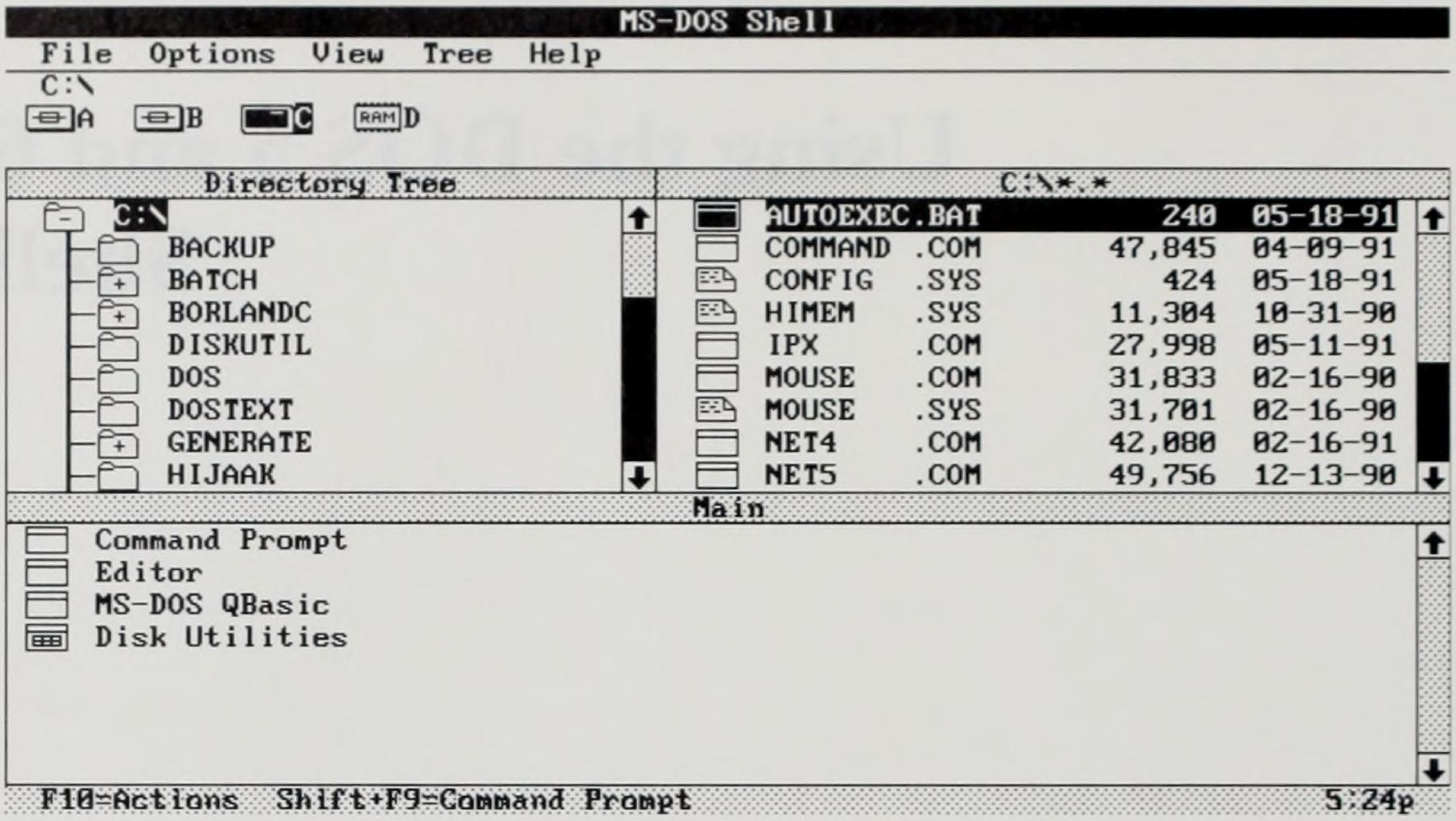
Throughout this book, you have issued your commands from the DOS prompt or using DOS batch files. Beginning with DOS 4, DOS introduced a graphical, menu-driven interface that lets you run programs or perform file operations by clicking on files with your mouse or by selecting menu options. Although the DOS shell program changed between versions 4 and 5, the shell did not change between versions 5 and 6. This chapter gets you up and running with the DOS 5 and 6 shell. In Chapter 40 you will examine the shell's advanced features.

Starting the DOS 5 and 6 Shell

To start the DOS 5 and 6 shell, type **DOSSHELL** at the DOS prompt, just as you would execute any DOS command.

```
C:\> DOSSHELL <ENTER>
```

Many users place the DOSSHELL in their AUTOEXEC.BAT file to start the shell automatically each time their system boots. The shell displays the screen shown in Figure 39-1.



```
                                MS-DOS Shell
 File  Options  View  Tree  Help
 C:\
 A  B  C  D
        Directory Tree                                  C:\*.*
 C:\                                   AUTOEXEC.BAT           240  05-18-91
   BACKUP                              COMMAND  .COM       47,845  04-09-91
   BATCH                               CONFIG   .SYS          424  05-18-91
   BORLANDC                            HIMEM    .SYS       11,304  10-31-90
   DISKUTIL                            IPX      .COM       27,998  05-11-91
   DOS                                 MOUSE    .COM       31,833  02-16-90
   DOSTEXT                             MOUSE    .SYS       31,701  02-16-90
   GENERATE                            NET4     .COM       42,080  02-16-91
   HIJAAK                              NET5     .COM       49,756  12-13-90
                                    Main
 Command Prompt
 Editor
 MS-DOS QBasic
 Disk Utilities

 F10=Actions  Shift+F9=Command Prompt                                    5:24p
```

Figure 39-1. *The DOS 5 and 6 shell interface*

If your computer has a mouse, the DOS shell lets you use the mouse to select files and programs. Before you invoke the shell, however, you must first install the mouse device driver that accompanied your mouse on floppy disk. Depending on your mouse driver, you may need to use a CONFIG.SYS DEVICE entry (for a driver such as MOUSE.SYS) or invoke a memory-resident program from within your AUTOEXEC.BAT file (for a program such as MOUSE.COM).

The DOS shell consists of several parts. The title bar appears at the top of your screen and contains the words "MS-DOS Shell." Immediately below the title is the menu bar, which contains the File, Options, View, Tree, and Help options. To activate a specific menu, click on the menu using your mouse, press F10 to select the menu bar, and type the letter of the desired menu option that appears underlined; you can also hold down ALT and type the first letter of the menu you desire. When you select a menu, the shell will display a pull-down menu, as shown in Figure 39-2.

To select a menu option, click on the option with your mouse, type the letter appearing underlined in the option name, or use the arrow keys to highlight the option and press ENTER. If a menu option appears dim, the option is not currently available. If the option is followed by three periods (called ellipses), the shell will display a dialog box when you select the option to request additional information. Several menu options work as toggles, meaning the first time you select the option, the shell enables an operation, and the second time you select the option, the shell disables the operation. For such options, the shell will display a diamond next to the option name when the operation is enabled. Several menus may display a keypress or key combination, to the right of an option, called a *hot key*, which is a keypress

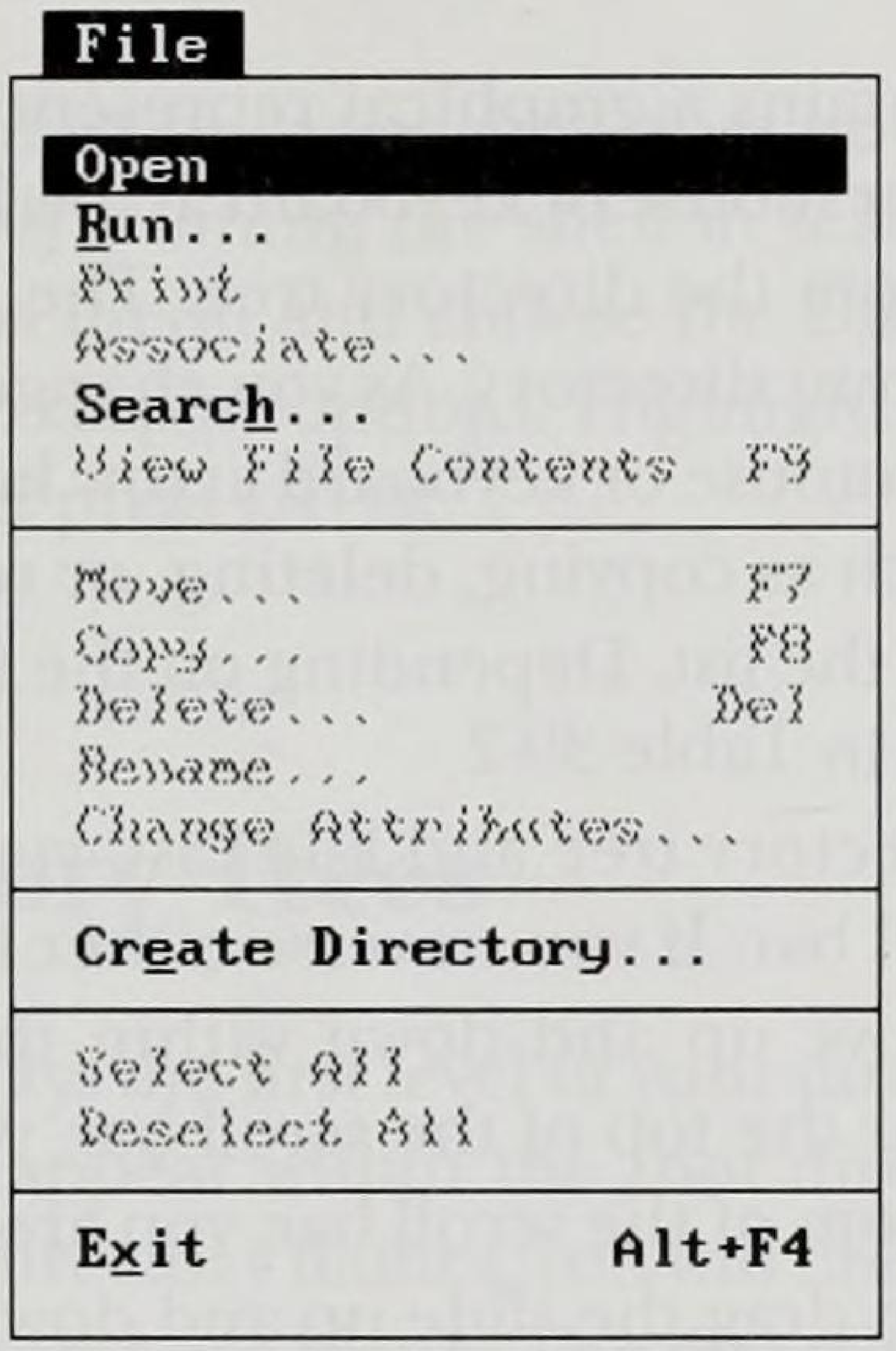

File
Open
Run...
Print
Associate...
Search...
View File Contents F9
Move... F7
Copy... F8
Delete... Del
Rename...
Change Attributes...
Create Directory...
Select All
Deselect All
Exit Alt+F4

Figure 39-2. *The File menu*

shortcut to the option. You can press the combination to select the option without having to traverse the menu. If you choose not to select a menu option, press ESC or click the mouse outside of the menu.

The disk drive icons correspond to the drives from which the shell can access files or run programs. Depending on your computer's disk types, you may have one or more of the disk drive icons shown in Table 39-1.

To change drives, you can click on the desired icon with your mouse, hold down CTRL and press the letter corresponding to the desired drive, or repeatedly press TAB

Icon	Disk Drive Type
A	Floppy disk drive
C	Hard disk drive
RAM D	RAM drive
NET F	Network drive
CD	CD-ROM

Table 39-1. *The Disk Drive Icons*

until the disk icons are selected, and use your arrow keys to highlight the drive you desire and press ENTER.

The directory tree contains a graphical representation of the directory tree for the current disk. Using your mouse or keyboard arrow keys, you can select the current directory by selecting it from the directory tree. The File List displays the names of files that reside in the current directory. As you change directories, the shell updates the list of files. Using your mouse or keyboard arrow keys, you can select one or more files for file operations such as copying, deleting, or renaming, or you can even run a program that appears in the list. Depending on the file's type, the shell will display one of the file icons listed in Table 39-2.

If you examine the directory tree and File List, you will find a vertical bar to the right of each called a scroll bar. If you traverse either list with your keyboard's arrow keys, a small slide will move up and down within the list indicating your relative position. If the slide is near the top of the scroll bar, you are near the top of the list. If the slide is near the bottom of the scroll bar, you are near the end of the list. If you are using a mouse, you can drag the slide up and down the scroll bar to traverse the list. To drag the slide, aim your mouse pointer at the slide and press the mouse button. With the button depressed, move your mouse to drag the slide. When the slide is in the desired position, release the mouse button.

The Main Group area contains program groups or programs you can execute by double-clicking on the program name or highlighting the name with your arrow keys and pressing ENTER. In Chapter 40 you will learn how to add your own programs to the Main Group.

At the bottom of your screen the shell displays a status bar that defines commonly used key definitions and displays the current system time.

If you are using a mouse, you can click directly on disk icons, directories, filenames, or programs within the Main Group. If you are using your keyboard, you can press TAB to move from one area on the screen to another and then use your arrow keys to highlight the object, pressing ENTER to select it. If you need to move in the opposite direction, press SHIFT-TAB.

Icon	**File Type**
▭	Executable EXE, COM, or BAT file
🗎	Nonexecutable file

Table 39-2. *The File Icons*

Displaying the Shell in Graphics Mode

If your monitor supports graphics but your shell does not match the shell displayed in the book, you are very likely running the shell in text mode. To select graphics mode, select the shell Options menu and choose the Display option. The shell will display a list of different video display options. Highlight the graphics mode option that best fits your monitor and press ENTER.

Working with Directory Trees

By default, the shell only displays the first level of your directory tree, in other words, the directories whose names appear within the root directory. If you examine the icons that appear next to the directory names, you will find some icons contain a plus sign, some a minus sign, and some are blank. The directories whose icon is blank do not contain additional directories. The directories whose icons contain a plus sign have additional levels of directories that are not currently displayed. A directory whose icon contains a minus sign has all of its next-level directories visible.

Using your mouse or keyboard, you can select a directory and *expand* (make visible) or *collapse* (remove from view) one or more levels of directories beneath the directory.

Expanding the Next Level of Directories

The shell provides you with several ways to expand a directory to its next level of directories. First, you can click on the plus sign that appears in the directory's icon with your mouse. If you are using your keyboard, use your arrow keys to highlight the directory and press the plus sign or invoke the Tree menu and select the Expand One Level option. Note the plus sign to the right of the option indicating the option's hot key.

Expanding a Directory's Tree

To expand all the directories beneath a directory, click on the directory name with your mouse or use your arrow keys to select the directory. Next, type an asterisk (*). To expand your entire directory tree, you can select the root directory and type * or you can press the CTRL-* key combination, regardless of the current directory.

Collapsing a Directory's Branches

To collapse the branches beneath a directory, click on the minus sign that appears in the directory's icon or highlight the entry with your arrow keys and press the minus sign. The minus sign is the hot key corresponding to the Tree menu's Collapse Branch option.

Exiting the DOS Shell

There are two ways to exit the DOS shell. First, if you have finished using the shell and want to return to the DOS prompt, you can press ALT-F4 or select the File menu and choose the Exit option. DOS will remove the shell from memory. To use the shell again, you must reissue the DOSSHELL command.

At other times, you may only want to temporarily exit the shell to issue one or two DOS commands. Because you may have several programs active, each of which has open files, ending each program to quit the shell, only to restart the shell later, would be inconvenient and frustrating. Using SHIFT-F9, you can temporarily exit the shell to DOS, leaving your active programs unchanged. When you press this key combination, the shell will display the DOS prompt letting you issue commands. After you issue your last command, type **EXIT** to return to the shell.

```
C:\DOS> EXIT <ENTER>
```

The shell will redisplay its screen, letting you resume your work right where you left off.

WARNING *Never turn off your computer or restart DOS when you have only temporarily exited the shell to DOS. If you have programs active within the shell with open files, changes to the files will be lost. As a rule, always use ALT-F4 to exit permanently from the shell to DOS before turning your computer off.*

If while you have temporarily exited the shell to DOS, you created directories or files, your changes may not appear in the directory tree or File List when you later return to the shell. To update the directory list, select the desired directory and press CTRL-F5.

Working with Files

The shell's File menu provides a series of options that let you print, move, copy, delete, or rename selected files. Before you can use these options, you must first select one or more files.

Selecting Files

Several file commands, such as copy, print, and delete, let you work with one or more files. To select one file with your mouse, click on the filename. To select one file with your keyboard, highlight the file's name with your arrow keys. Using either technique, the shell will display the selected file's icon in reverse video. If you inadvertently select the wrong file, click on the correct file or highlight the correct file with your keyboard arrow keys.

To select two or more files, the steps you must perform differ, depending on whether the files are a consecutive group or dispersed throughout the File List. To select a group of consecutive files with your mouse, click on the name of the first file in the list. Hold down SHIFT and click on the name of the last file in the list. The shell will display the entire group of files in reverse video. To select a group of consecutive files with your keyboard, use the arrow keys to highlight the first file in the group. Hold down SHIFT and use the arrow keys to highlight remaining files in the group.

To select two or more files that aren't consecutive, hold down CTRL while you click on each file. To select the files with your keyboard, use the arrow keys to select the first file you desire. Press SHIFT-F8. The shell will display the word "Add" in the status bar next to the current time. Use the arrow keys to highlight the second file desired. Press SPACEBAR to select the file. Repeat these steps to highlight and select the desired files. After all files are selected, press SHIFT-F8 again.

If you want to select all the files in the current directory, you can use the File menu's Select All option. Likewise, to cancel your file selections, select the File menu and choose the Deselect All option.

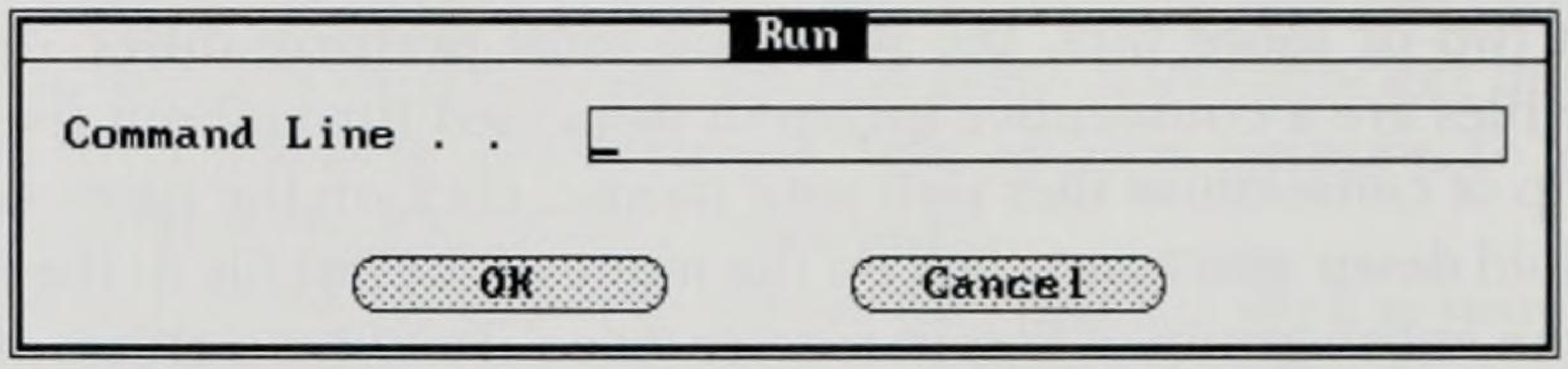
Selecting Files from Different Directories

There may be times when you want to copy, move, or delete files that reside in different directories. By default, the shell only lets you select files from one directory at a time. Using the Options menu's Select Across Directories option, you can enable and disable the selection of files from within one of several directories. When you enable this option you can select one or more files from the current directory, change directories, and then follow the procedure for selecting two or more nonconsecutive files. When the option is enabled, the shell displays a diamond next to the option name. When the option is disabled, the diamond is not displayed.

Running Programs

The shell provides several ways for you to run application programs and DOS commands. First, you can highlight a COM, EXE, or BAT file within the File List and press ENTER to execute the file. If you are using a mouse, you can double-click on the filename. Second, the shell lets you associate a file type (extension) with a specific program. For example, you might associate files with the DOC extension to your word processor. When you highlight an associated file and press ENTER or double-click on the filename, DOS will execute the associated program, loading the data file specified. The File menu's Open option lets you run a highlighted program or the program associated with a highlighted file.

The File menu's Run option lets you type in a command, just as you would from the DOS prompt. When you select the Run option the shell will display the Run dialog box.



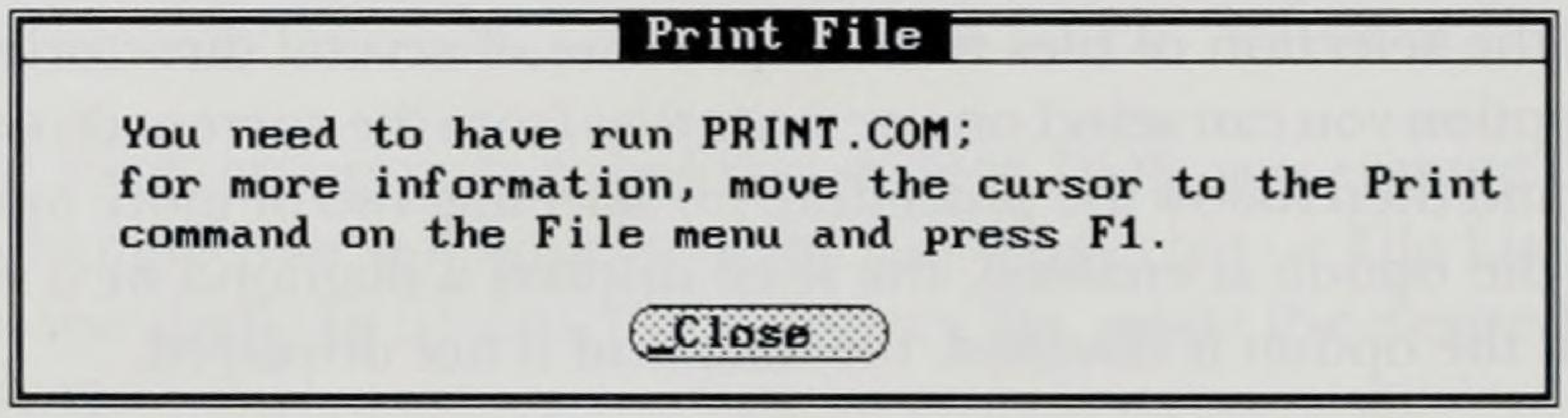
Command Line . . _

Type in the command name and its command-line arguments. The shell will run the command, displaying the command's output on its own screen. When the command completes, the shell will prompt you to press any key to continue before redisplaying the shell screen.

Last, as discussed later in this chapter, the shell lets you run programs by selecting the program from the Main Group that appears at the bottom of your screen.

Printing One or More Files

The File menu's Print option lets you print one or more selected files. When you select this option, the shell will briefly display a dialog box informing you it has queued the files for printing. Before you can use the shell's Print option, you must first run the PRINT command, as discussed in Chapter 14. If you have not yet run PRINT and you select the Print option, the shell will display the following dialog box telling you PRINT.COM has not been loaded and the files cannot print:

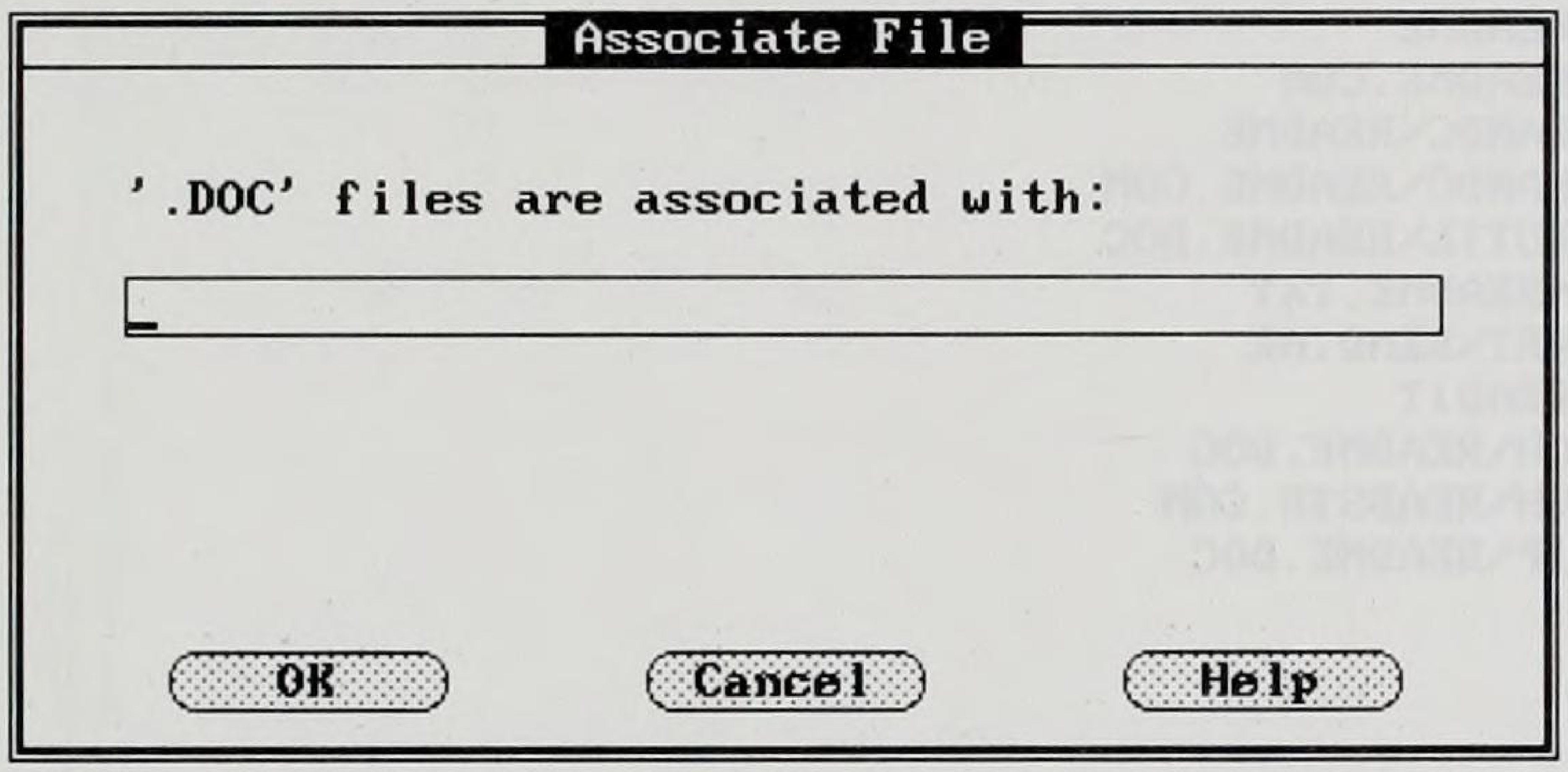


You need to have run PRINT.COM;
for more information, move the cursor to the Print
command on the File menu and press F1.

If this dialog box appears, exit the shell and invoke the PRINT command, as discussed in Chapter 14.

Associating a Program with a File Type

The shell lets you associate a file with a program. For example, you might associate TXT files with the DOS editor. When you later select the file, the shell will automatically run the associated program, loading the selected file. To associate a file type to a program, select a file of the desired type, invoke the File menu, and choose the Associate option. The shell will display this dialog box, prompting you to type in the program name to which you want the files associated.



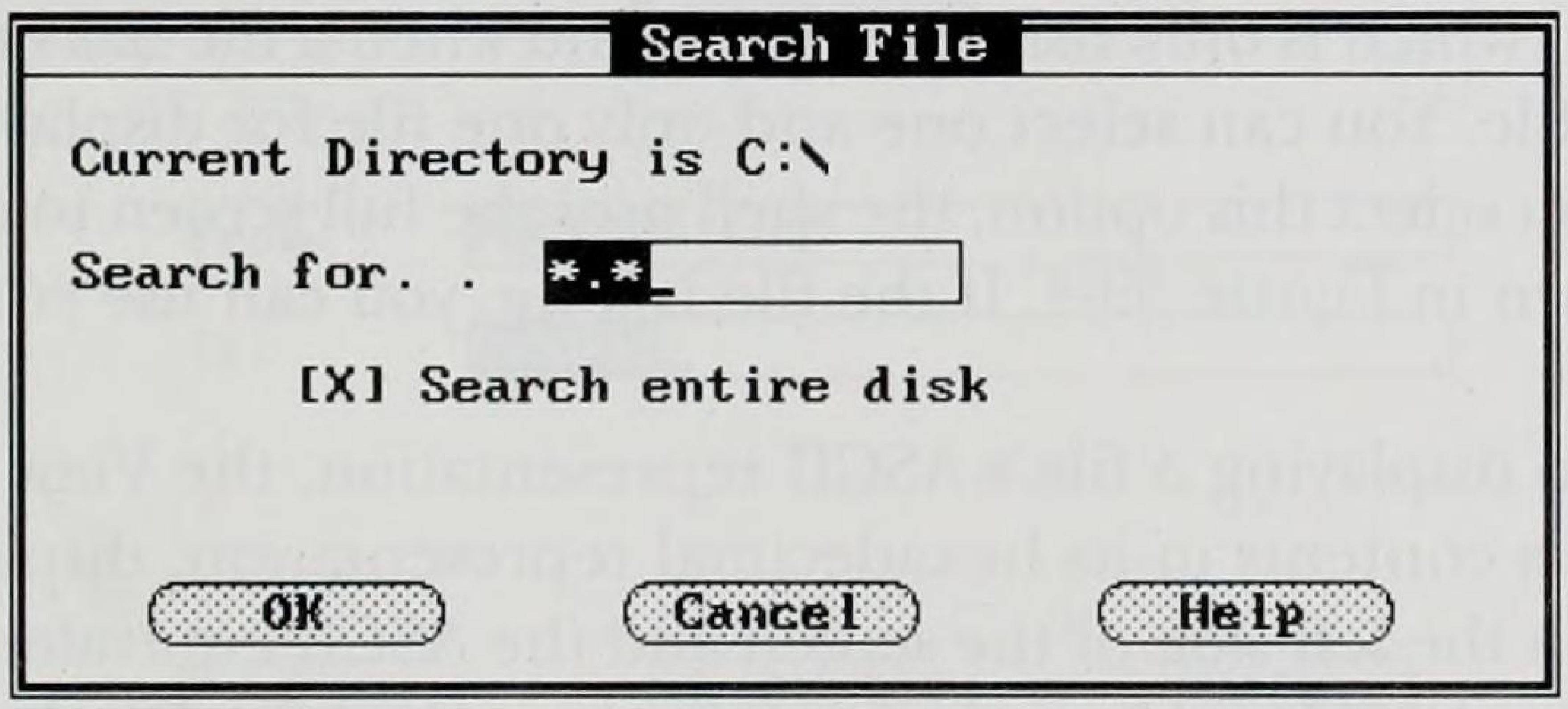
'.DOC' files are associated with:

Type in the complete path name to the program and press ENTER.

To remove an association at a later time, follow the same steps, removing the program name from the dialog box. If you select the program from which you want to dissociate files and then select the File menu's Associate option, the shell will display a dialog box similar to the Associate File box, but that prompts you to enter the file type (extension) you want to dissociate from the program.

Searching Your Disk for Specific Files

If you misplace a file on your disk, you can use the File menu's Search option to quickly locate it. When you select this option, the shell displays the Search File dialog box, prompting you to type in the name of the file to locate.

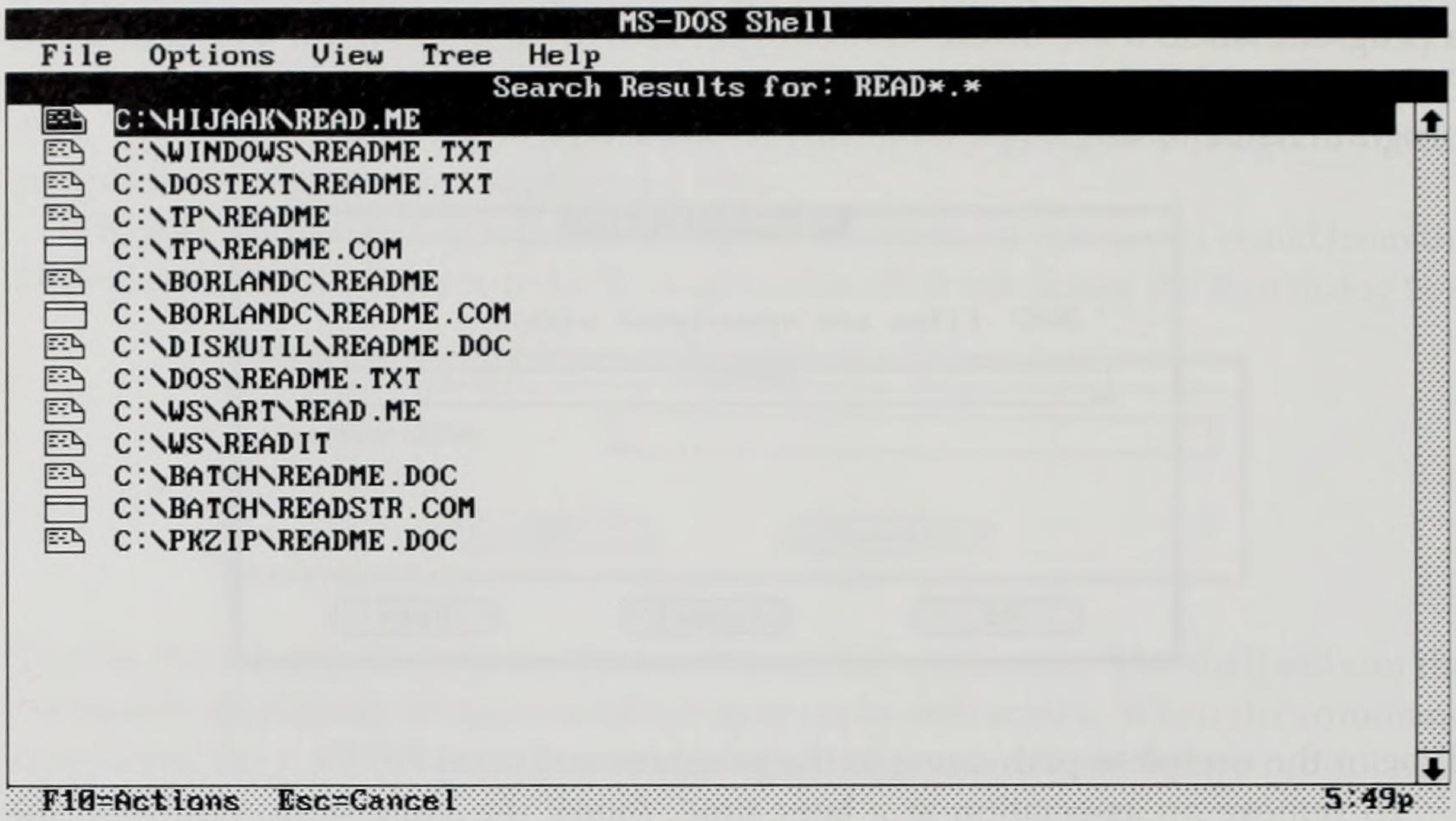


Current Directory is C:\

Search for. . *.*

[X] Search entire disk

Type in the name of the desired file. The search operation fully supports the DOS wildcards. When you press ENTER or click on the OK option, the shell will begin searching your disk for matching files. If the search operation is successful, the shell will display the directory path to each matching file, as shown in Figure 39-3. From this matching list you can select one or more files for a file operation or to run a program. To return to the shell, press ESC.



```
                         Search Results for: READ*.*
C:\HIJAAK\READ.ME
C:\WINDOWS\README.TXT
C:\DOSTEXT\README.TXT
C:\TP\README
C:\TP\README.COM
C:\BORLANDC\README
C:\BORLANDC\README.COM
C:\DISKUTIL\README.DOC
C:\DOS\README.TXT
C:\WS\ART\READ.ME
C:\WS\READIT
C:\BATCH\README.DOC
C:\BATCH\READSTR.COM
C:\PKZIP\README.DOC
```

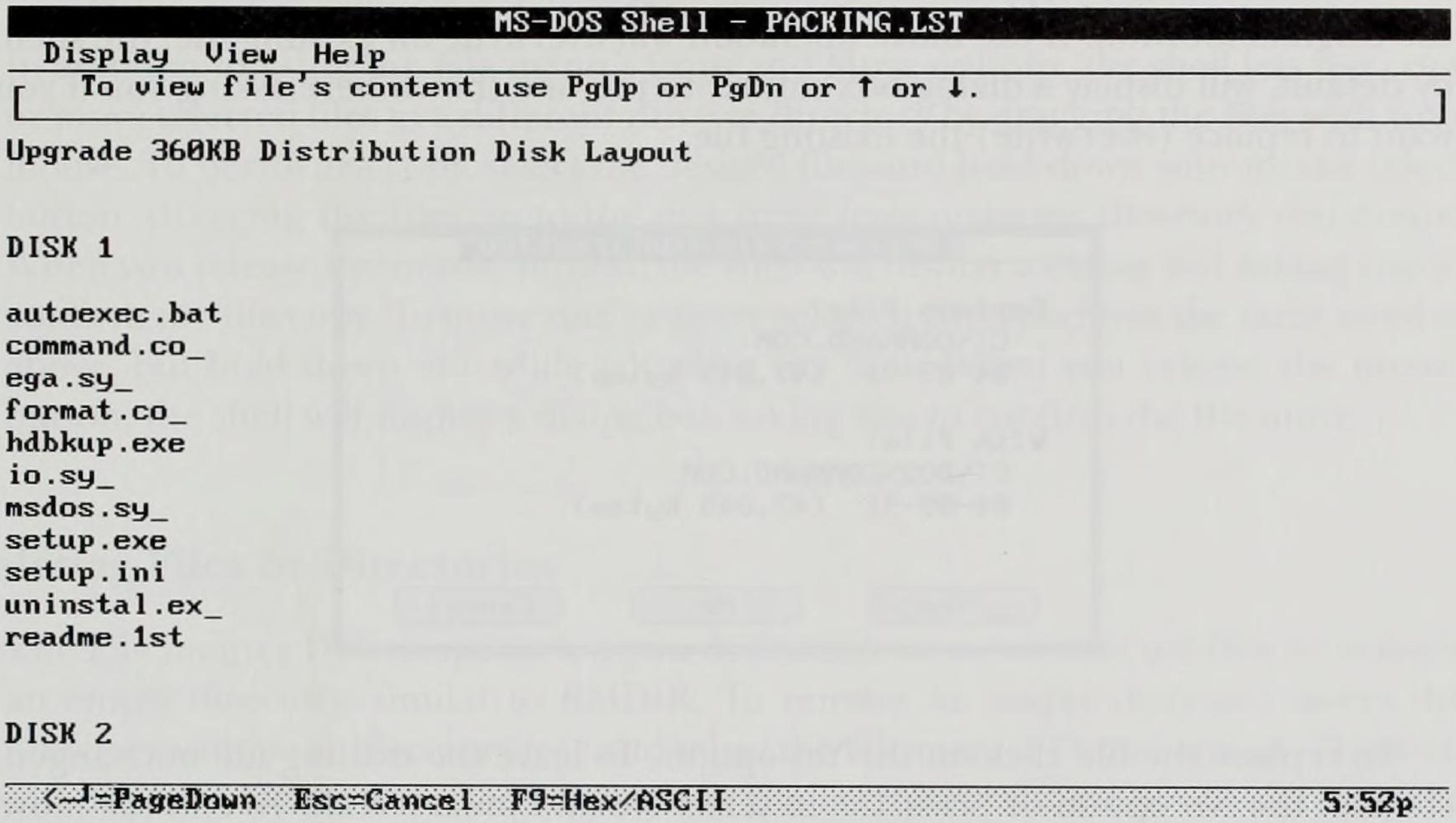
Figure 39-3. *Search File's list of matching files*

Viewing a File's Contents

In Chapter 8 you learned that the TYPE command lets you display an ASCII file's contents to your screen. Within the shell, you can use the File menu's View File Contents option (which is only visible on the menu when a file has been selected) to page through a file. You can select one and only one file for display using the View option. When you select this option, the shell uses the full screen to display the file's contents, as shown in Figure 39-4. If the file is long, you can use PGUP and PGDN to traverse the file.

In addition to displaying a file's ASCII representation, the View option lets you examine the file's contents in its hexadecimal representation, displaying the hexadecimal values on the left side of the screen and the ASCII equivalents on the right. To switch between ASCII and hexadecimal display, select the Display menu's ASCII and Hex options. To return to the shell, press ESC.

The View File Contents option will not let you change the file. To edit a file use the DOS editor or your word processor.

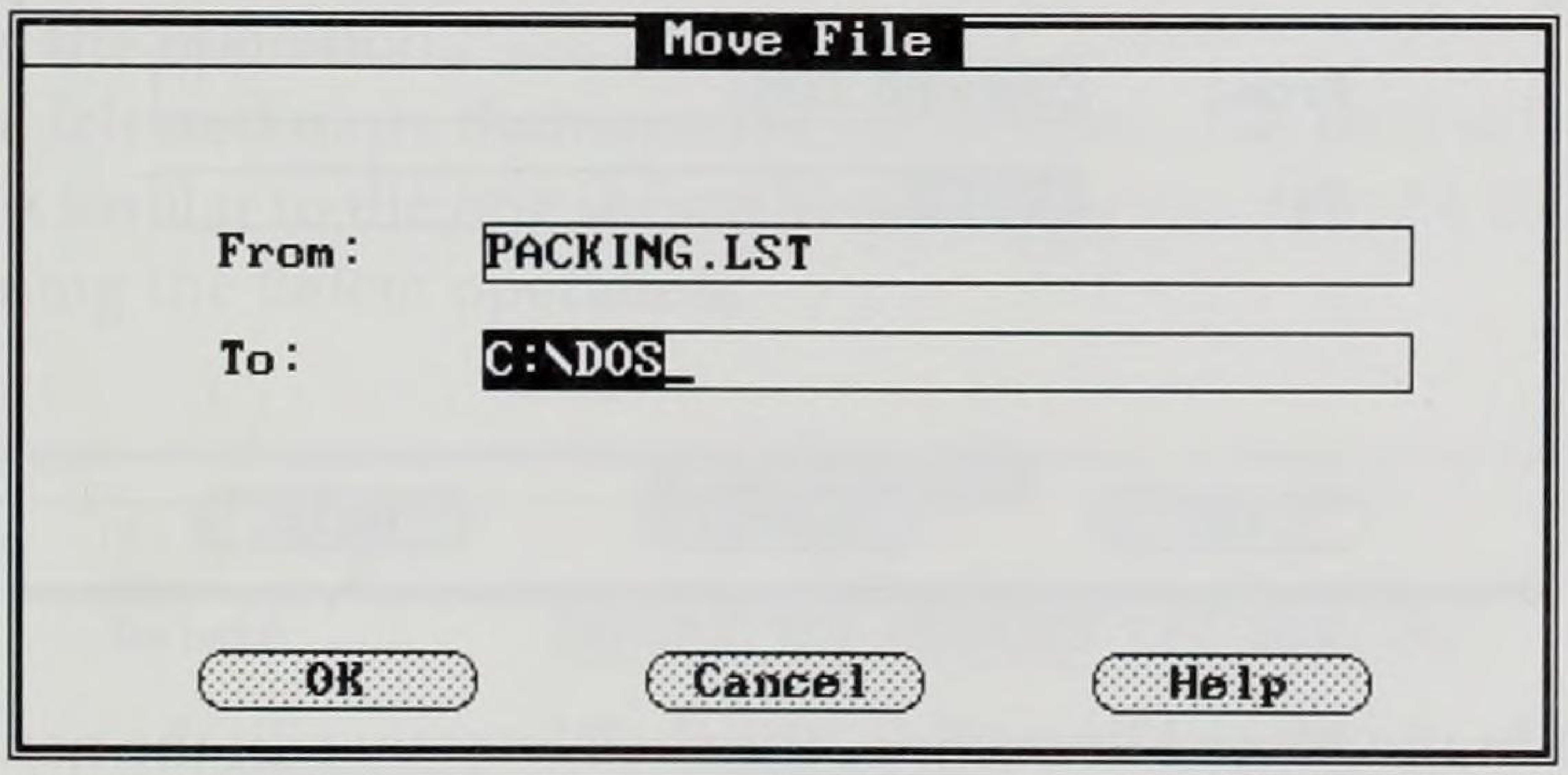

MS-DOS Shell - PACKING.LST
Display View Help
<—┘=PageDown Esc=Cancel F9=Hex/ASCII
5:52p

```
To view file's content use PgUp or PgDn or ↑ or ↓.

Upgrade 360KB Distribution Disk Layout


DISK 1

autoexec.bat
command.co_
ega.sy_
format.co_
hdbkup.exe
io.sy_
msdos.sy_
setup.exe
setup.ini
uninstal.ex_
readme.1st


DISK 2
```

Figure 39-4. *Viewing an ASCII file*

Moving One or More Files

The File menu's Move option (visible on the menu only after a file has been selected) lets you move one or more selected files from one drive or directory to another. When you select the Move option, the shell will display the following dialog box, letting you view the names of the selected files and possibly add more, and prompting you to type in the target drive or directory.

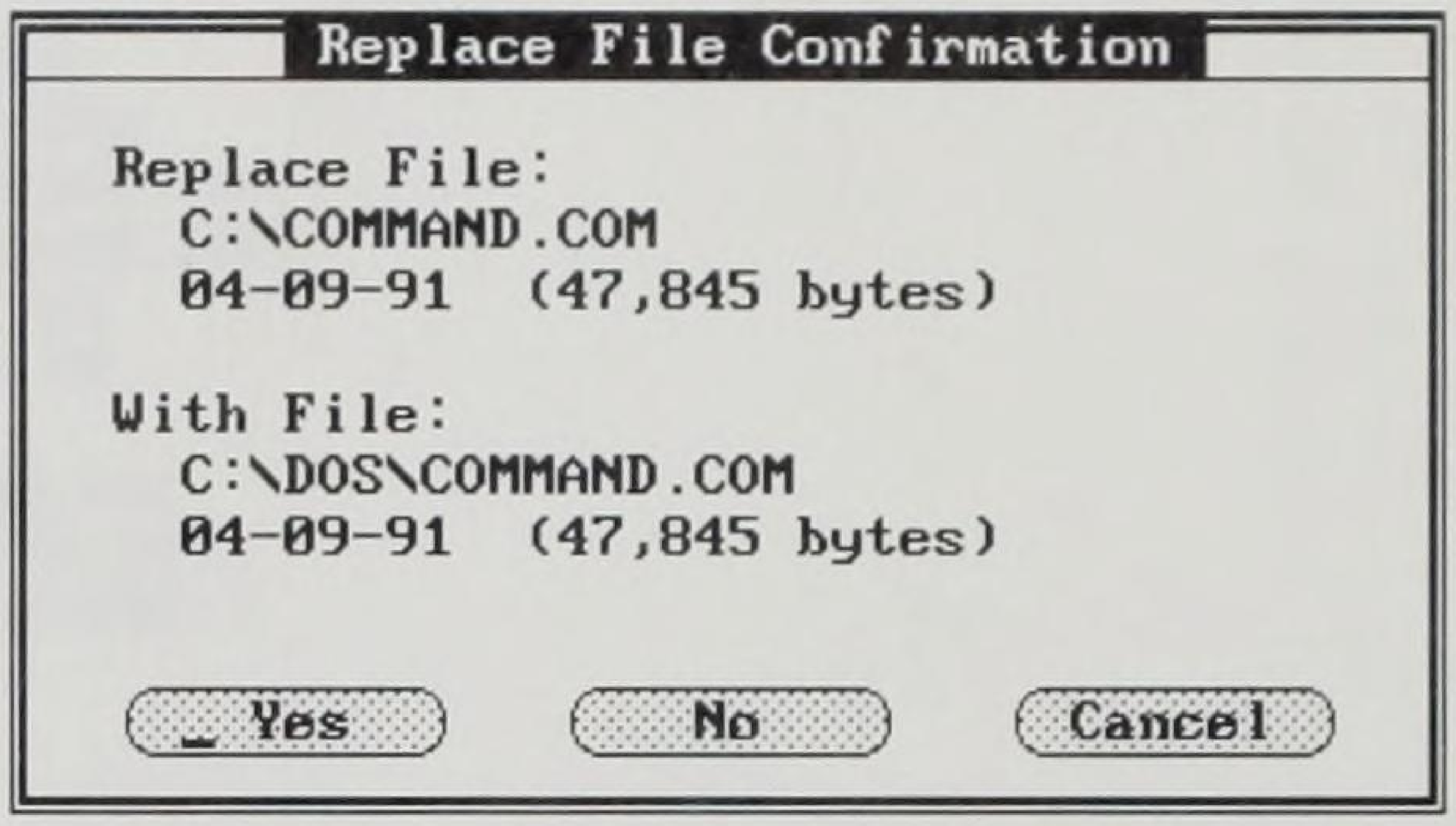


From: PACKING.LST

To: C:\DOS

To add or delete files to or from the list, click on the From box with your mouse or press TAB to highlight the box. Edit the filenames as desired. Next, type in the name of the target drive or directory and press ENTER or click on the OK option with your mouse.

The shell will first copy the files to the new location and then delete the files from the original location. If the move operation will overwrite an existing file, the shell, by default, will display a dialog box similar to the one shown here asking you if you want to replace (overwrite) the existing file.

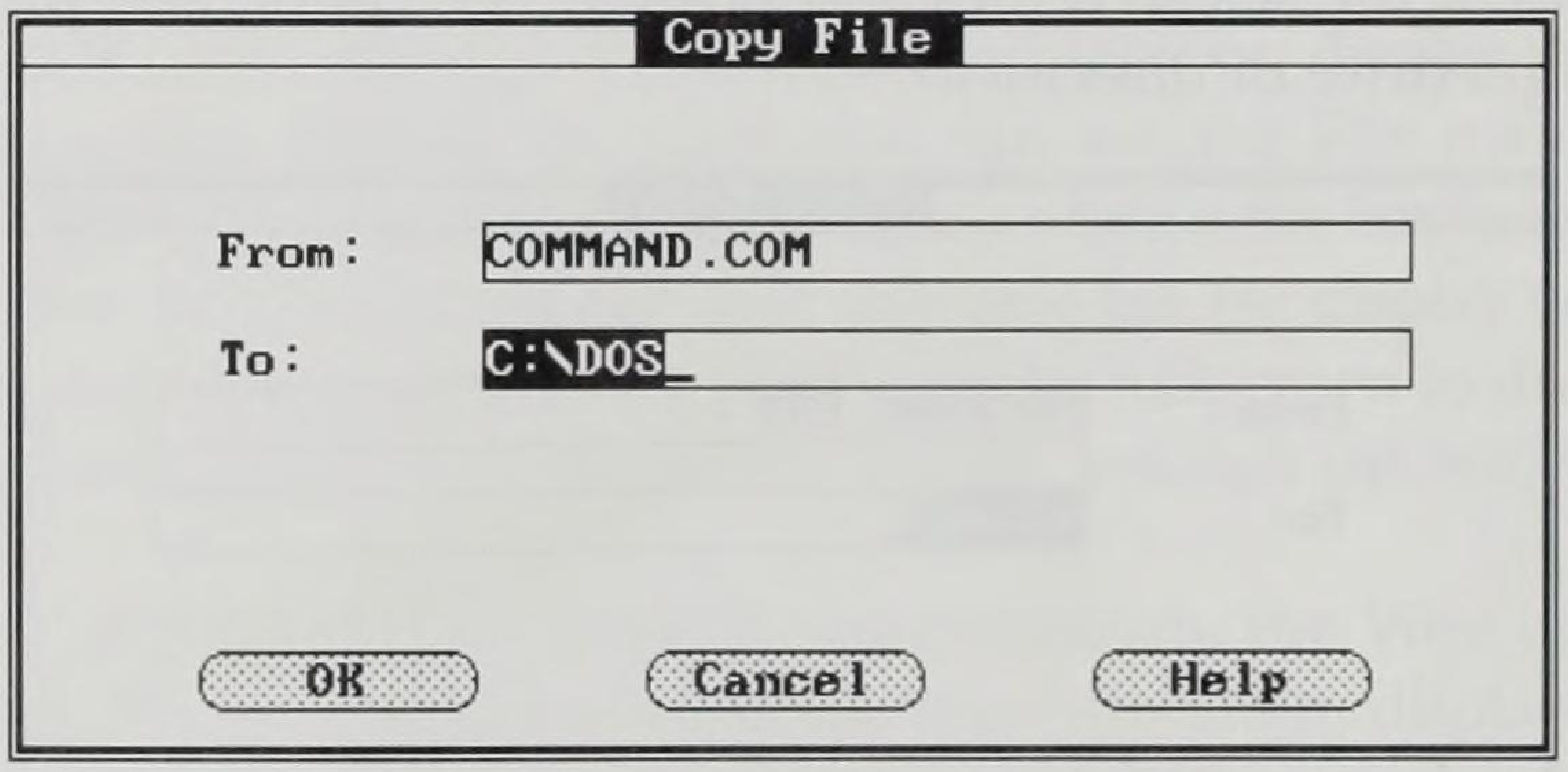


```
Replace File Confirmation

Replace File:
  C:\COMMAND.COM
  04-09-91  (47,845 bytes)

With File:
  C:\DOS\COMMAND.COM
  04-09-91  (47,845 bytes)
```

To replace the file click on the Yes option. To leave the existing file unchanged, click on the No option or press TAB to select No and press ENTER. To end the move operation, select Cancel.

Copying One or More Files

The File menu's Copy option lets you copy one or more selected files from one disk or directory to another. When you invoke the Copy operation, the shell will display a Copy File dialog box similar to the one shown here, which prompts you to type in the target drive or directory:



```
Copy File

From:   COMMAND.COM
To:     C:\DOS_
```

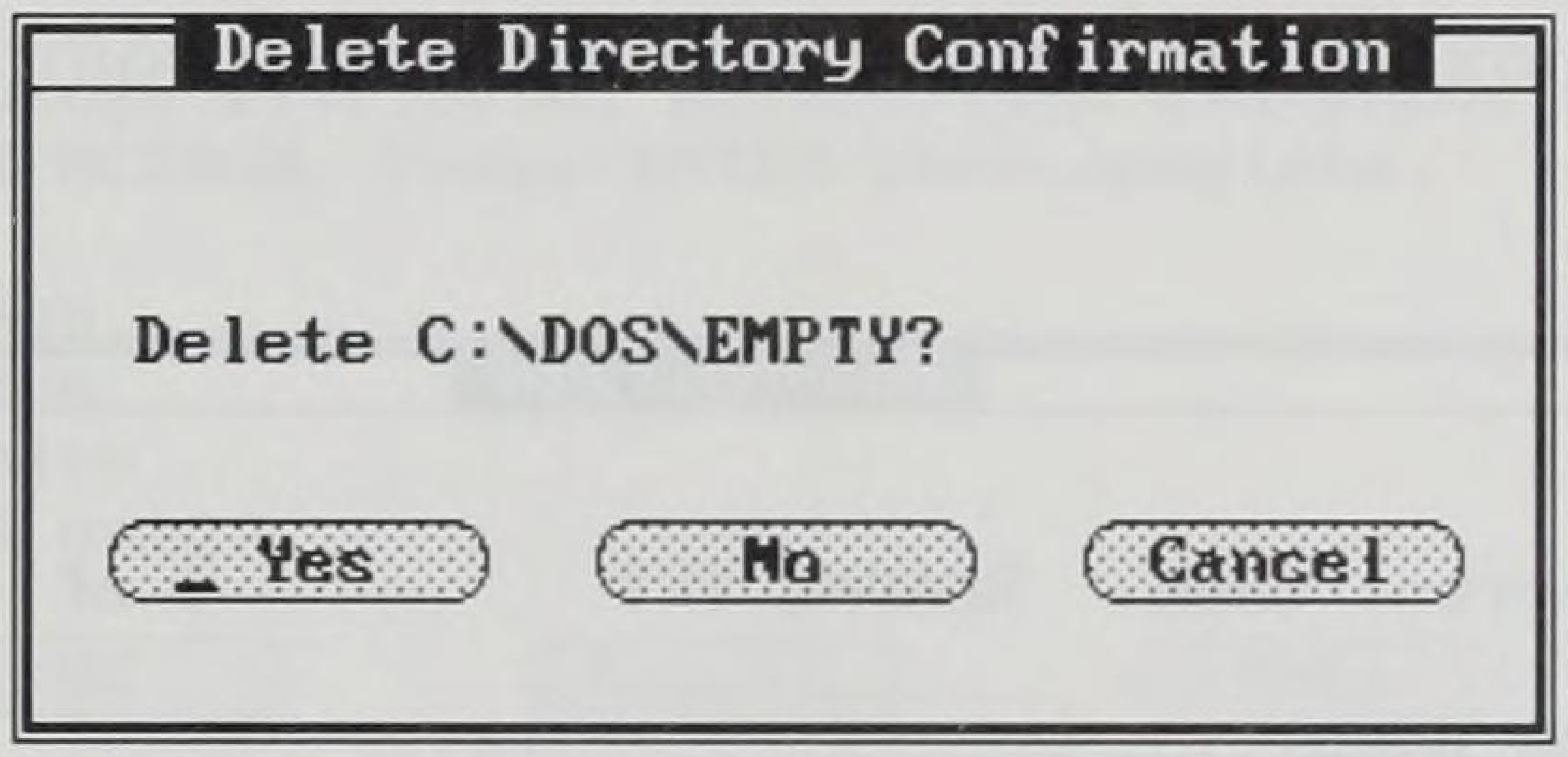
As you just read in the Move File section, the shell lets you edit the names of the files selected for copying. Type in the target drive or directory and press ENTER, or click on the OK option. If a selected file will overwrite an existing file on the target drive or directory, the shell will display a dialog box similar to the Replace File Confirmation box that asks you if you want to replace (overwrite) the existing file. As discussed in the Move File section, select Yes to overwrite the file, No to leave the existing file unchanged, and Cancel to end the Copy operation.

Moving or Copying Files by Dragging with the Mouse

In addition to using the File menu's Copy and Move options, the shell lets you copy or move selected files to a different drive or directory by dragging the files with your mouse. To perform a copy, select the desired files and hold down your mouse select button, dragging the files on to the disk drive icon or target directory you desire. When you release the mouse button, the shell will display a dialog box asking you to confirm the file copy. To move one or more selected files, perform the same steps as above, but hold down ALT while you drag the files. When you release the mouse button, the shell will display a dialog box asking you to confirm the file move.

Deleting Files or Directories

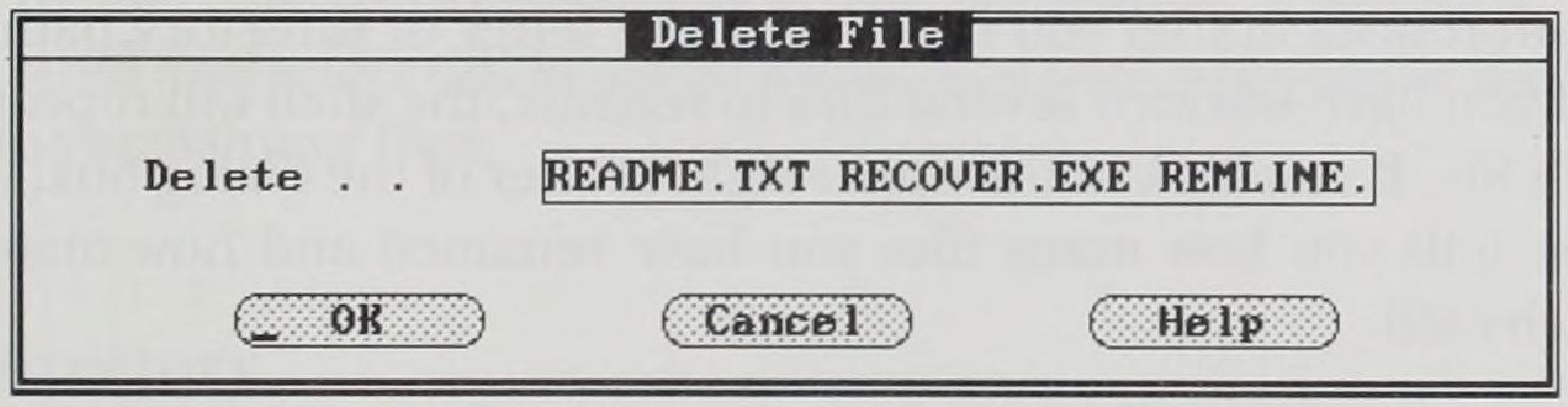
The File menu's Delete option lets you delete one or more selected files or remove an empty directory, similar to RMDIR. To remove an empty directory, select the directory within the directory tree and select the File menu's Delete option. The shell will display a Delete Directory dialog box similar to one shown here, asking you to confirm the directory removal:



Delete C:\DOS\EMPTY?

To remove the directory select Yes. To leave the directory on disk select No. Select Cancel to end the operation.

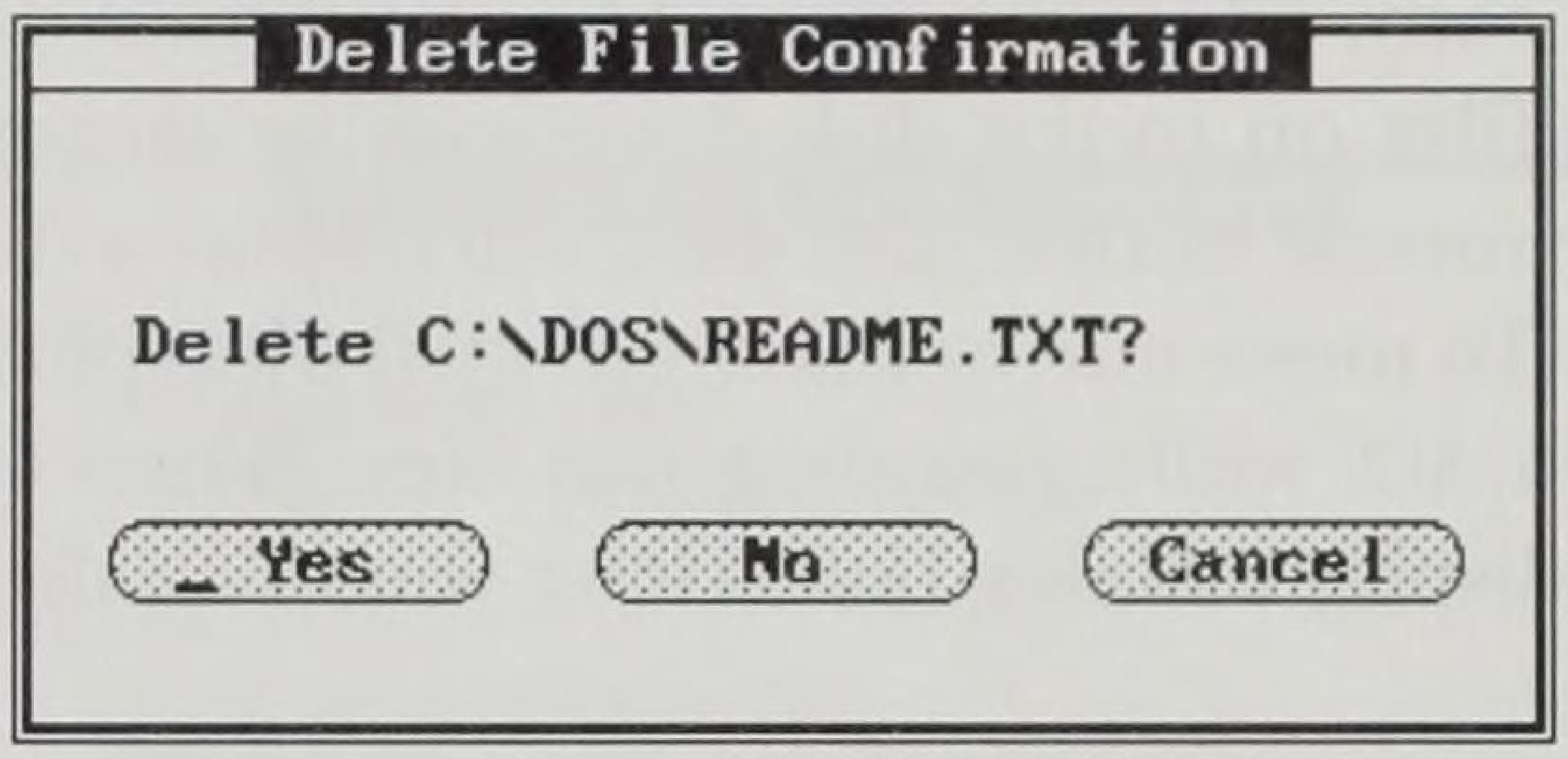
If you have selected more than one file for deletion, the shell will display a Delete File dialog box similar to the one shown here, letting you edit the list of selected files before beginning the delete operation:



Delete . . README.TXT RECOVER.EXE REMLINE.

Select OK to continue the delete operation.

By default, the shell will individually prompt you to confirm that you want to delete each selected file by displaying a Delete File Confirmation dialog box similar to the one shown here:

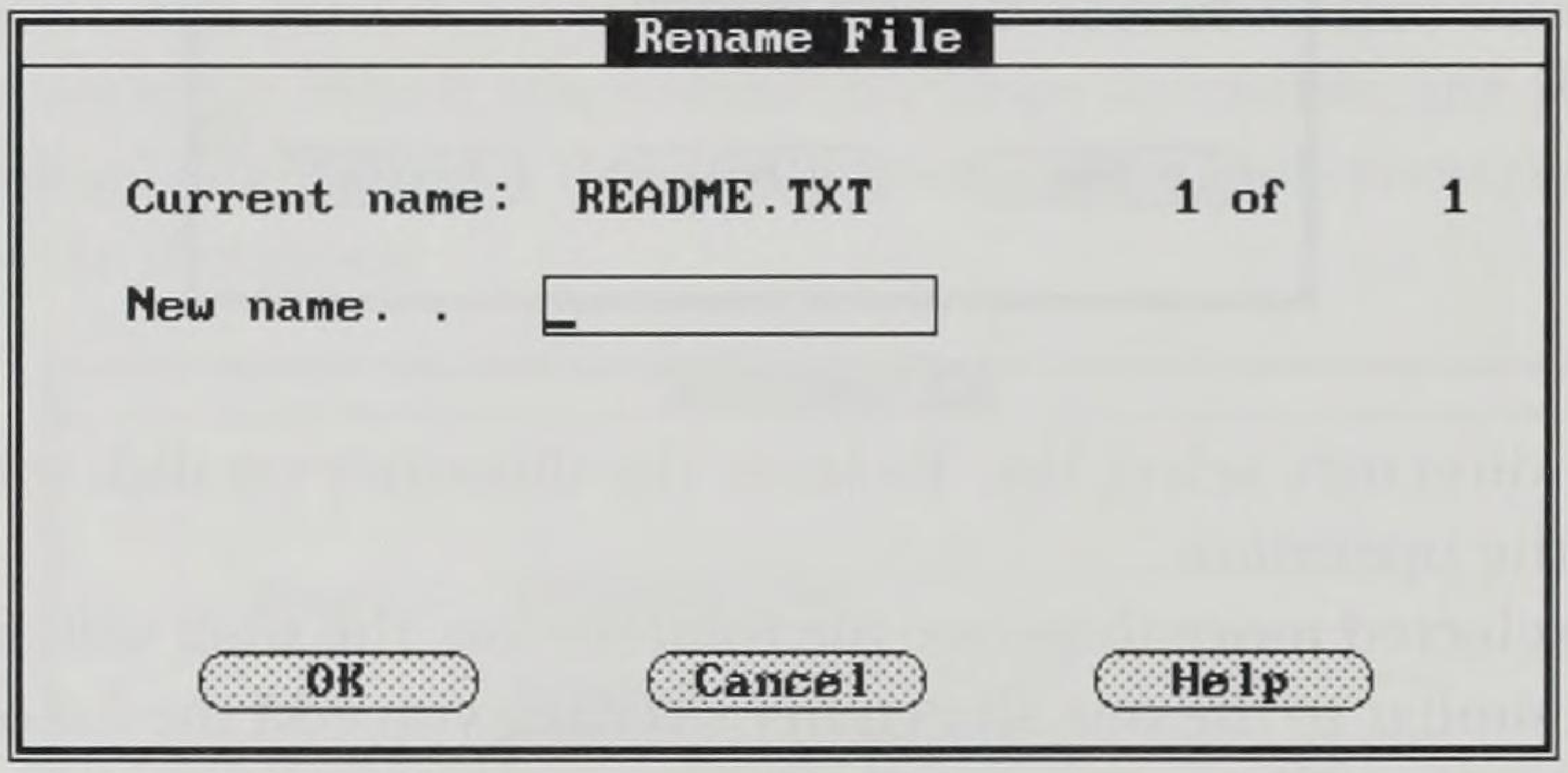


Delete C:\DOS\README.TXT?

To delete the file, select Yes. To leave the file on disk, select No. If you have selected multiple files, you can end the delete operation by selecting Cancel.

Renaming One or More Files

The File menu's Rename option (visible only when a file has been selected) lets you rename one or more selected files. When you select the Rename option, the shell will display a dialog box similar to the one shown here, prompting you for the desired filename:



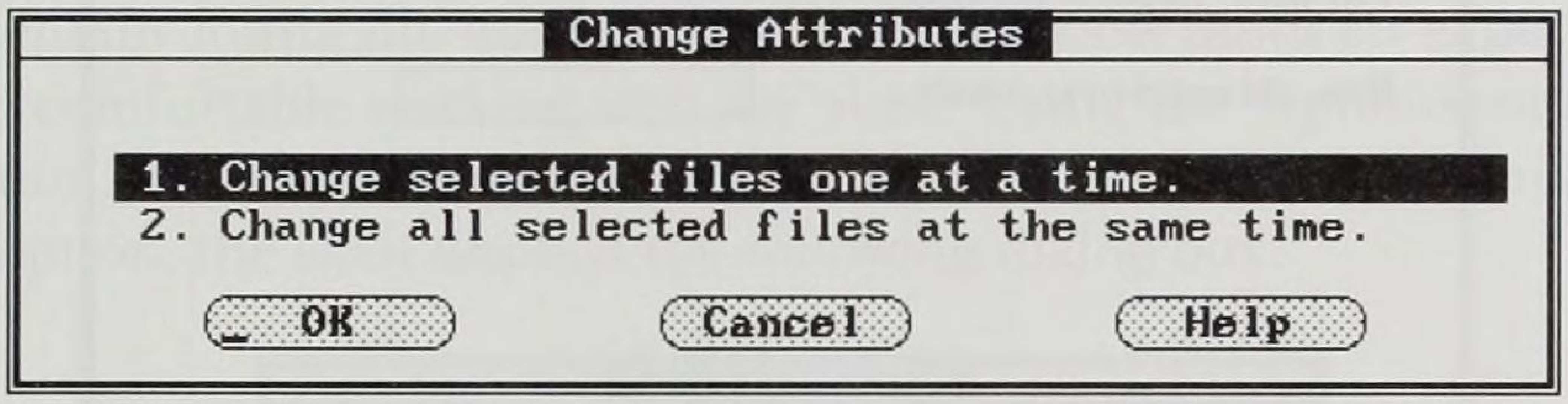
Current name: README.TXT 1 of 1

New name. .

Type in the new filename you desire. Like the RENAME command, the shell's Rename option does not let you include a drive letter or directory path in the new filename. If you have selected several files to rename, the shell will repeat this dialog box for each file. If you look in the upper-right corner of the dialog box, you will find a count that tells you how many files you have renamed and how many files were originally selected.

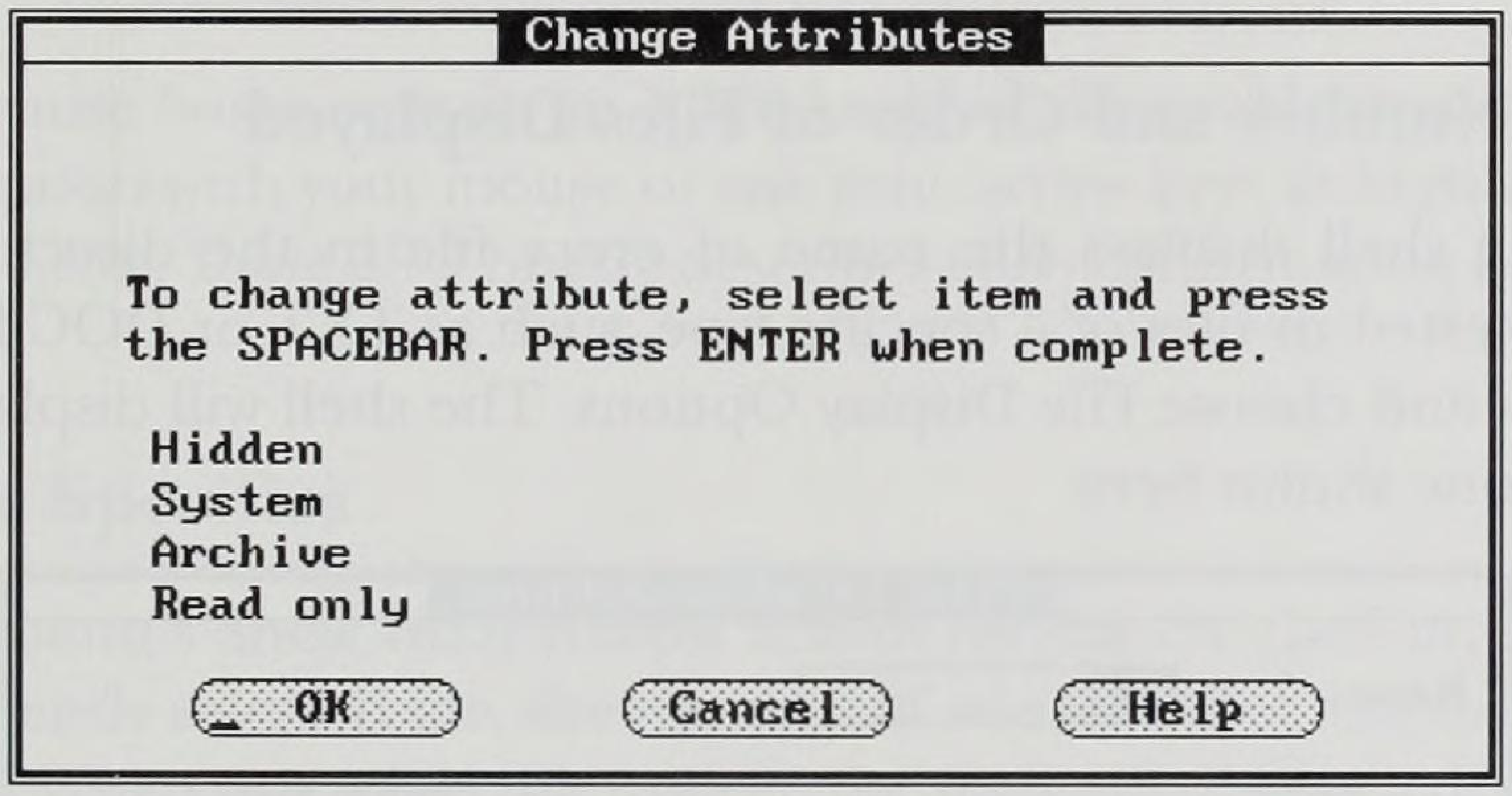
Changing Attributes of One or More Files

Just as the ATTRIB command lets you set or clear a file's archive, read-only, hidden, and system attributes, so too will the File menu's Change Attributes command (visible only when a file has been selected). When you have selected several files and you choose this option, the shell will first display the dialog box shown here asking you if you want to set attributes for selected files on an individual basis or if you want to set the attributes of all files at once.



1. Change selected files one at a time.
2. Change all selected files at the same time.

Select option 1 or 2 as you desire. If you select option 2 to set all the files' attributes at one time, the shell will display the following dialog box that lets you select the attributes you desire.



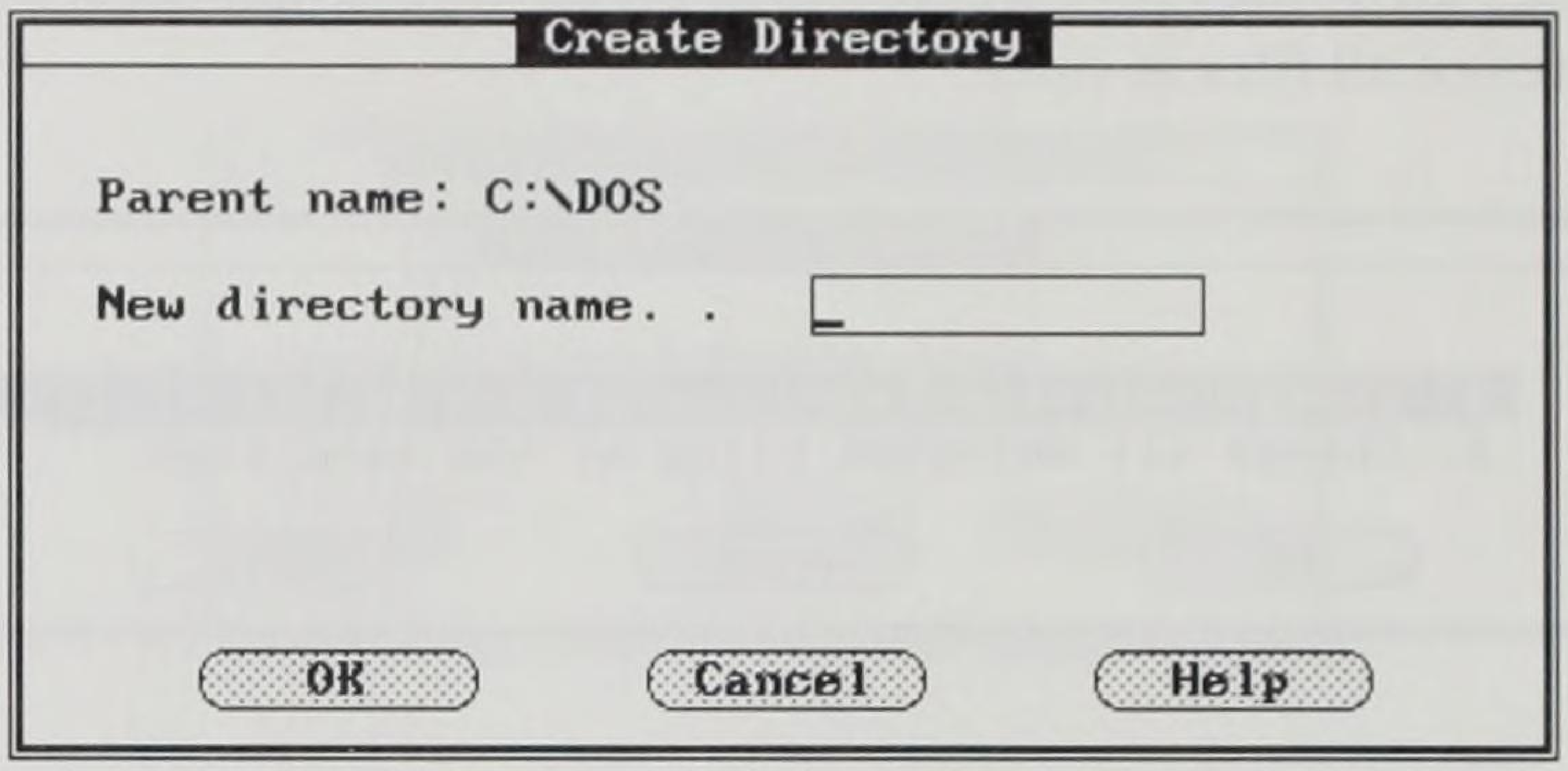
To change attribute, select item and press the SPACEBAR. Press ENTER when complete.

To select or remove an attribute, click on the attribute with your mouse or highlight the attribute and press SPACEBAR. The shell will display a triangle to the left of attributes it will set and nothing to the left of attributes it will clear. Select OK to assign the attributes.

If you want to individually set each file's attributes, the shell will display a dialog box similar to the Change Attributes box, but containing a filename. Select the attributes desired and select OK to assign them. The shell will display the dialog box for each of the remaining files.

Creating a Directory

The File menu's Delete option lets you remove empty directories from your disk. To add a directory, select the directory within which you want the new directory. Next, select the File menu and choose the Create Directory option. The shell will display a Create Directory dialog box similar to the one shown here, which displays the current directory and prompts you for the desired directory name:



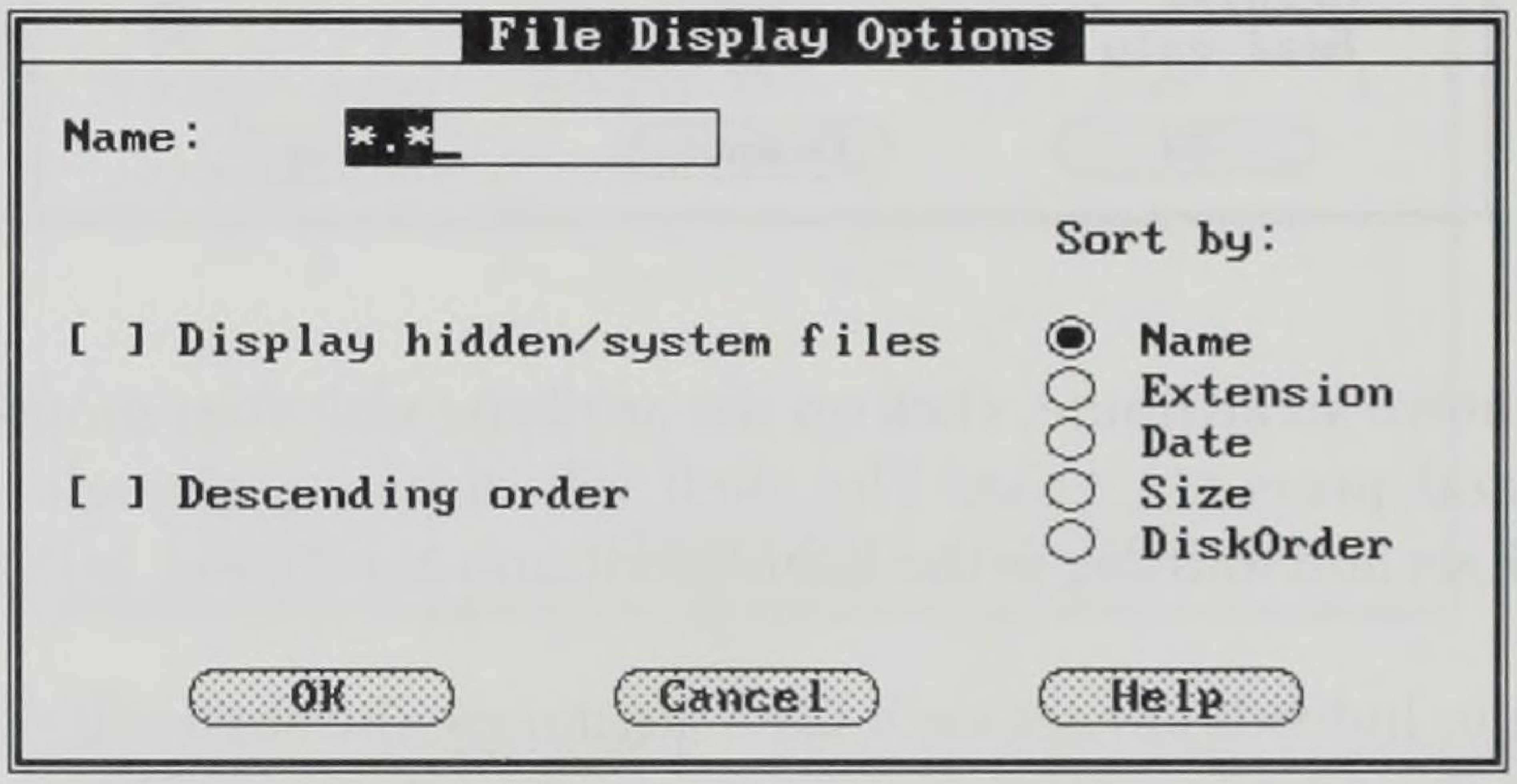
Parent name: C:\DOS

New directory name. . _

Type in the directory name desired. Like the MKDIR command, the Create Directory option won't let you create a directory whose name matches an existing file or directory name.

Changing the Number and Order of Files Displayed

By default, the shell displays the name of every file in the directory. If you are primarily interested in files of a specific type, such as TXT or DOC files, select the Options menu and choose File Display Options. The shell will display a dialog box similar to the one shown here:



Name: *.*_

[ ] Display hidden/system files

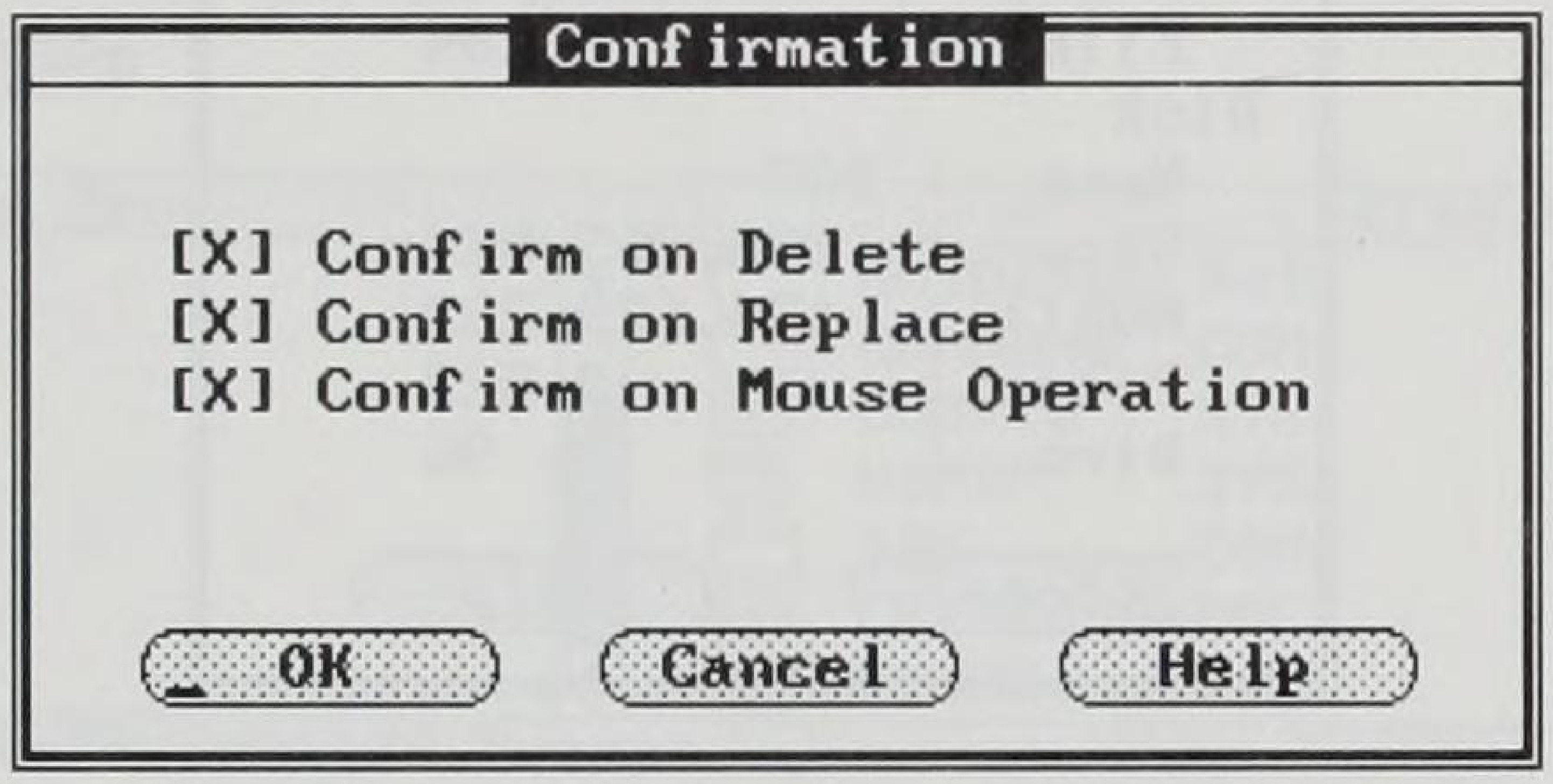
[ ] Descending order

Sort by:
● Name
○ Extension
○ Date
○ Size
○ DiskOrder

"Name" lets you type in the wildcard specification for the file type you want displayed. "Display hidden/system files" directs the shell to display files in the Files List whose system or hidden attribute is set. By default, the shell does not display these files. To enable the display of hidden and system files, click on the option box with your mouse or press TAB to select the option and SPACEBAR to set or clear the option. "Descending order" directs the shell to display the files sorted from highest to lowest. If the option is clear, the shell uses ascending order. If the option contains an X, the shell uses descending. "Sort by" specifies the field by which the shell sorts the files. To select a field, click on the field with your mouse or press TAB to highlight the Sort by option and select the desired field with your arrow keys.

Enabling and Disabling Confirmation Dialog Boxes

By default, each time you delete a file, attempt to overwrite an existing file, or drag the mouse to move or copy a file, the shell displays a dialog box to confirm that you really want to perform the operation. Although these confirmations may prevent you from accidentally losing file information, they may slow down an experienced user who is very comfortable working with the shell. Using the Options menu's Confirmation option, you can enable and disable these confirmation options. When you select this option, the shell displays the following dialog box.



[X] Confirm on Delete
[X] Confirm on Replace
[X] Confirm on Mouse Operation

The options whose boxes contain an X are enabled. To enable or disable an option, click on the option with your mouse or use your arrow keys to highlight the option and press SPACEBAR. Table 39-3 briefly describes each confirmation option.

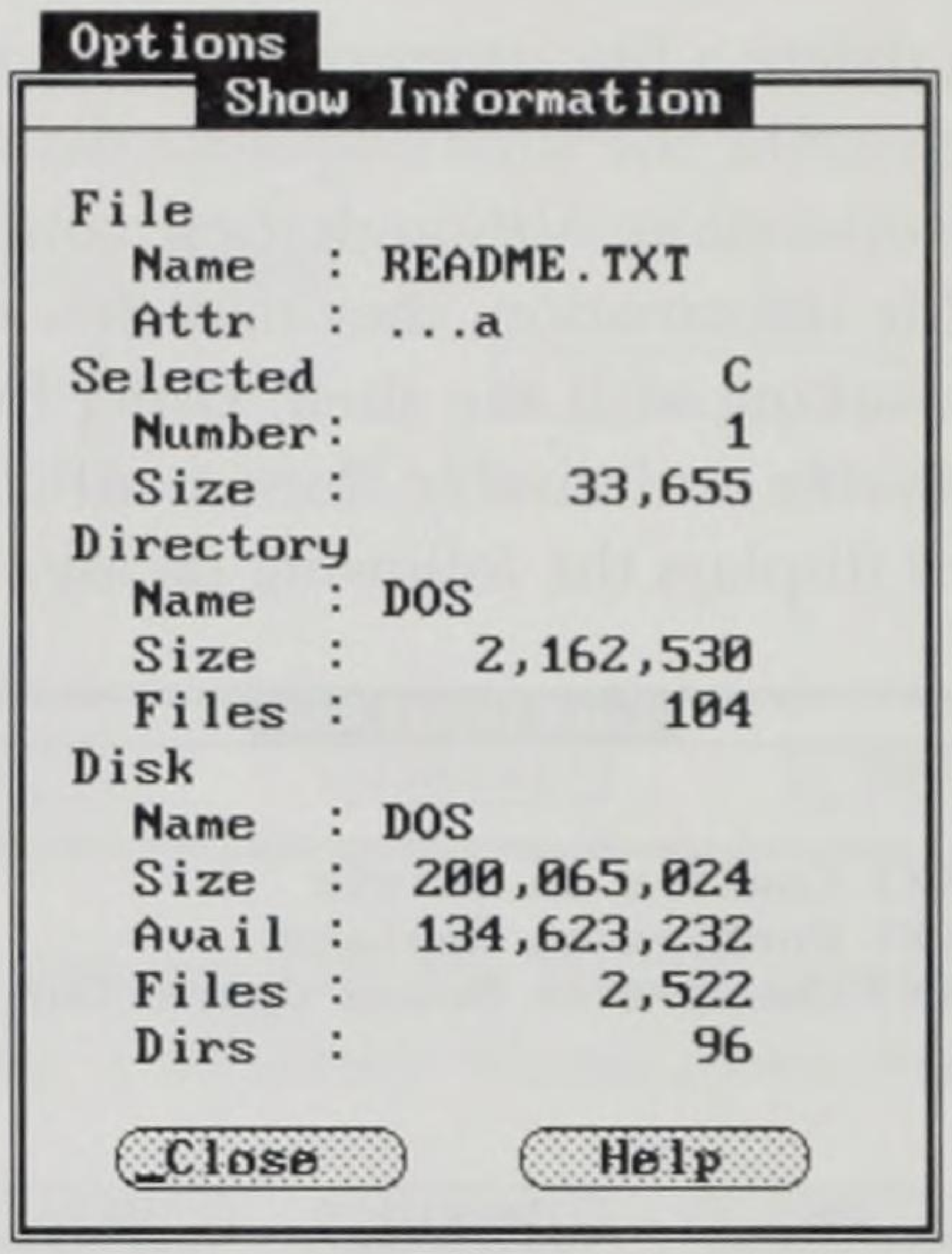
Displaying File Specifics

The Options menu's Show Information option directs the shell to display specifics about the currently selected file, the number of selected files, the current directory, and disk drive. When you select this option, the shell will display a dialog box similar to Figure 39-5.

"Name" specifies the last file selected. "Attr" specifies the file's attributes, where the letter a corresponds to the archive attribute, r to read-only, h to hidden, and s to system. "Selected" specifies the disk drive from which you have selected the files, the number of files, and their combined size. Using this value you can determine if a

Option	Confirmation
Confirm on Delete	Individually confirms each file deletion
Confirm on Replace	Confirms a move or copy operation that overwrites an existing file
Confirm on Mouse Operation	Confirms move or copy operations performed by a mouse drag

Table 39-3. The Shell Confirmation Options



```
File
   Name   : README.TXT
   Attr   : ...a
Selected              C
   Number:            1
   Size   :      33,655
Directory
   Name   : DOS
   Size   :   2,162,530
   Files  :         104
Disk
   Name   : DOS
   Size   :  200,065,024
   Avail  :  134,623,232
   Files  :       2,522
   Dirs   :          96
```

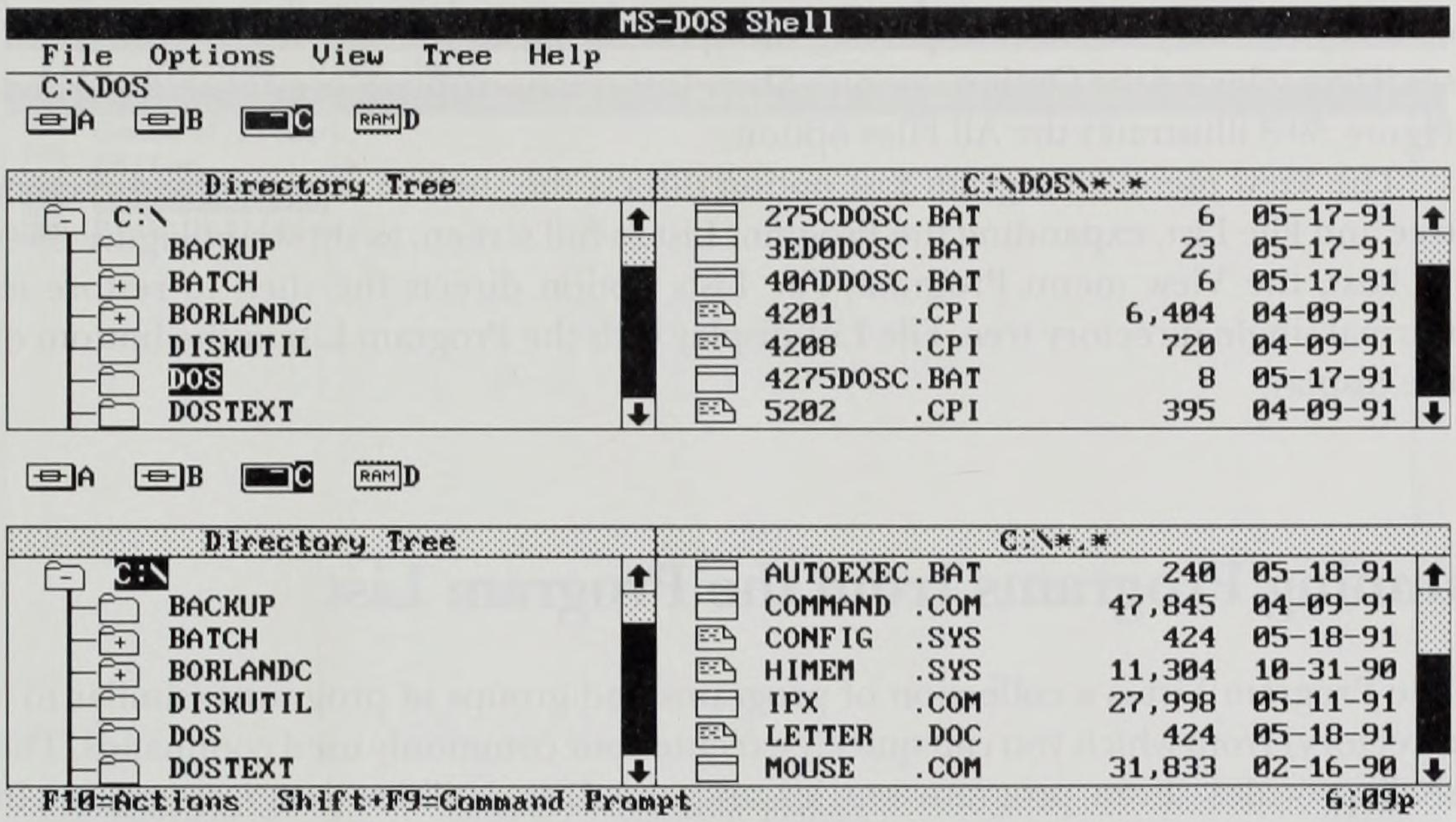
Figure 39-5. *The Show Information dialog box*

target disk has enough space to hold the selected files prior to a copy or move operation. If you have selected files from two disks, the shell will display a two-column summary. "Directory" specifies the name of the current directory, the number of files the directory contains, and their total size. "Disk" specifies the volume label for the current disk, the size of the disk, the available disk space, and the number of files and directories.

Changing the Shell Display

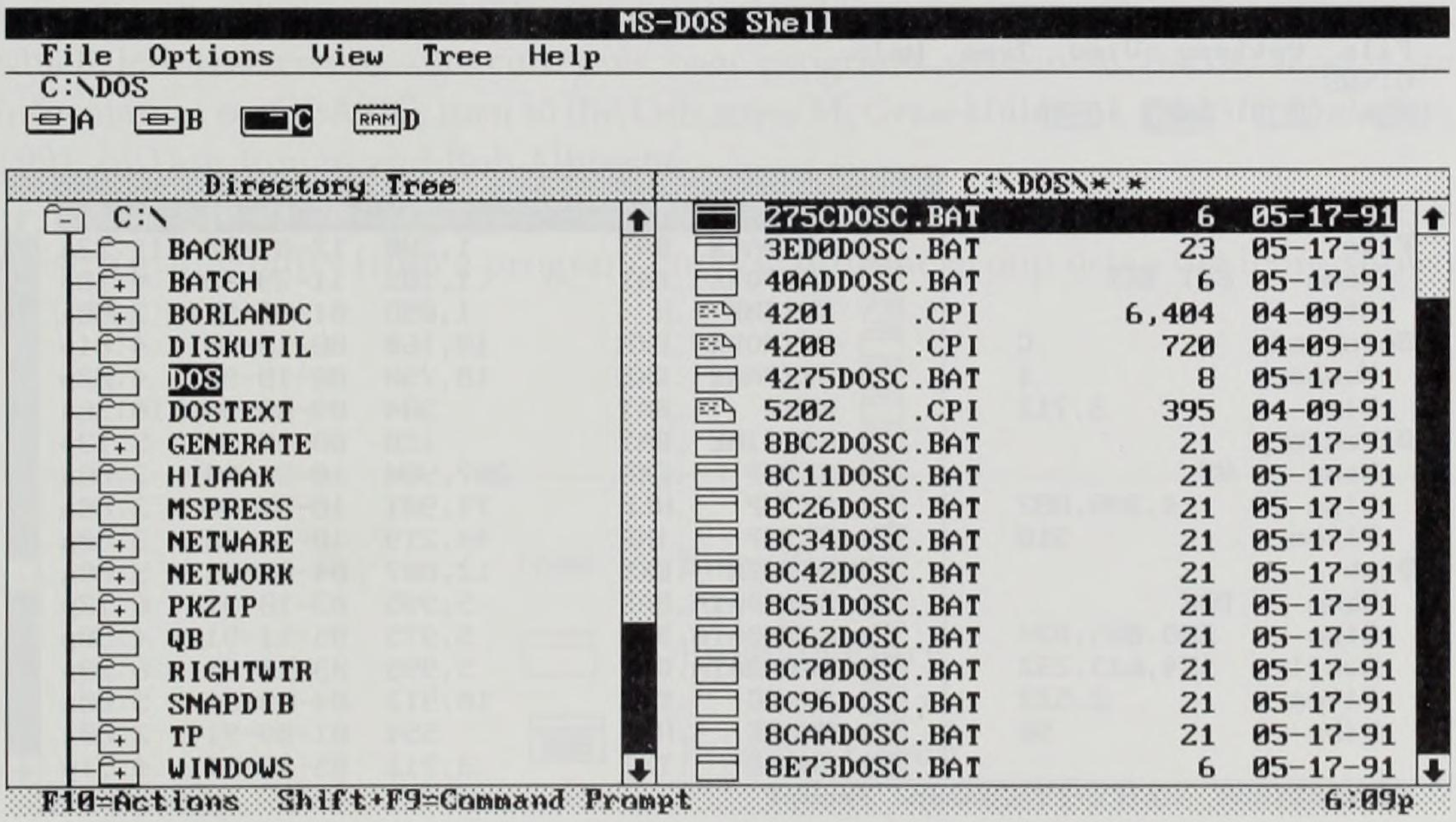
By default, the shell displays a single directory tree and File List. If you are copying or moving files from one disk or directory to another, you might want to display two directory trees and File Lists. To do so, select the View menu's Dual File Lists option. The shell will divide the screen in half, displaying two sets of disk drive icons, two directory trees, and two File Lists. Using the same steps you have used with one directory tree and File List, you can display two directory trees for the same disk or a different disk, as shown in Figure 39-6.

The View menu's Single File List option directs the shell to use the entire screen for a single directory tree and File List, removing the Program List, as shown in Figure 39-7.


MS-DOS Shell
File Options View Tree Help
C:\DOS
A B C D
Directory Tree
C:\DOS\*.*
Directory Tree
C:\*.*
F10=Actions Shift+F9=Command Prompt
6:09p

```
C:\                 275CDOSC.BAT       6  05-17-91
 ├BACKUP            3ED0DOSC.BAT      23  05-17-91
 ├BATCH             40ADDOSC.BAT       6  05-17-91
 ├BORLANDC          4201    .CPI   6,404  04-09-91
 ├DISKUTIL          4208    .CPI     720  04-09-91
 ├DOS               4275DOSC.BAT       8  05-17-91
 ├DOSTEXT           5202    .CPI     395  04-09-91

C:\                 AUTOEXEC.BAT     240  05-18-91
 ├BACKUP            COMMAND .COM  47,845  04-09-91
 ├BATCH             CONFIG  .SYS     424  05-18-91
 ├BORLANDC          HIMEM   .SYS  11,304  10-31-90
 ├DISKUTIL          IPX     .COM  27,998  05-11-91
 ├DOS               LETTER  .DOC     424  05-18-91
 ├DOSTEXT           MOUSE   .COM  31,833  02-16-90
```

Figure 39-6. Viewing dual File Lists


MS-DOS Shell
File Options View Tree Help
C:\DOS
A B C D
Directory Tree
C:\DOS\*.*
F10=Actions Shift+F9=Command Prompt
6:09p

```
C:\                 275CDOSC.BAT       6  05-17-91
 ├BACKUP            3ED0DOSC.BAT      23  05-17-91
 ├BATCH             40ADDOSC.BAT       6  05-17-91
 ├BORLANDC          4201    .CPI   6,404  04-09-91
 ├DISKUTIL          4208    .CPI     720  04-09-91
 ├DOS               4275DOSC.BAT       8  05-17-91
 ├DOSTEXT           5202    .CPI     395  04-09-91
 ├GENERATE          8BC2DOSC.BAT      21  05-17-91
 ├HIJAAK            8C11DOSC.BAT      21  05-17-91
 ├MSPRESS           8C26DOSC.BAT      21  05-17-91
 ├NETWARE           8C34DOSC.BAT      21  05-17-91
 ├NETWORK           8C42DOSC.BAT      21  05-17-91
 ├PKZIP             8C51DOSC.BAT      21  05-17-91
 ├QB                8C62DOSC.BAT      21  05-17-91
 ├RIGHTWTR          8C70DOSC.BAT      21  05-17-91
 ├SNAPDIB           8C96DOSC.BAT      21  05-17-91
 ├TP                8CAADOSC.BAT      21  05-17-91
 ├WINDOWS           8E73DOSC.BAT       6  05-17-91
```

Figure 39-7. Viewing a single File List

The View menu's All Files option directs the shell to display an alphabetized listing of every file on your disk, displaying the specifics about the current file you would see if you selected the Options menu's Show Information option previously discussed. Figure 39-8 illustrates the All Files option.

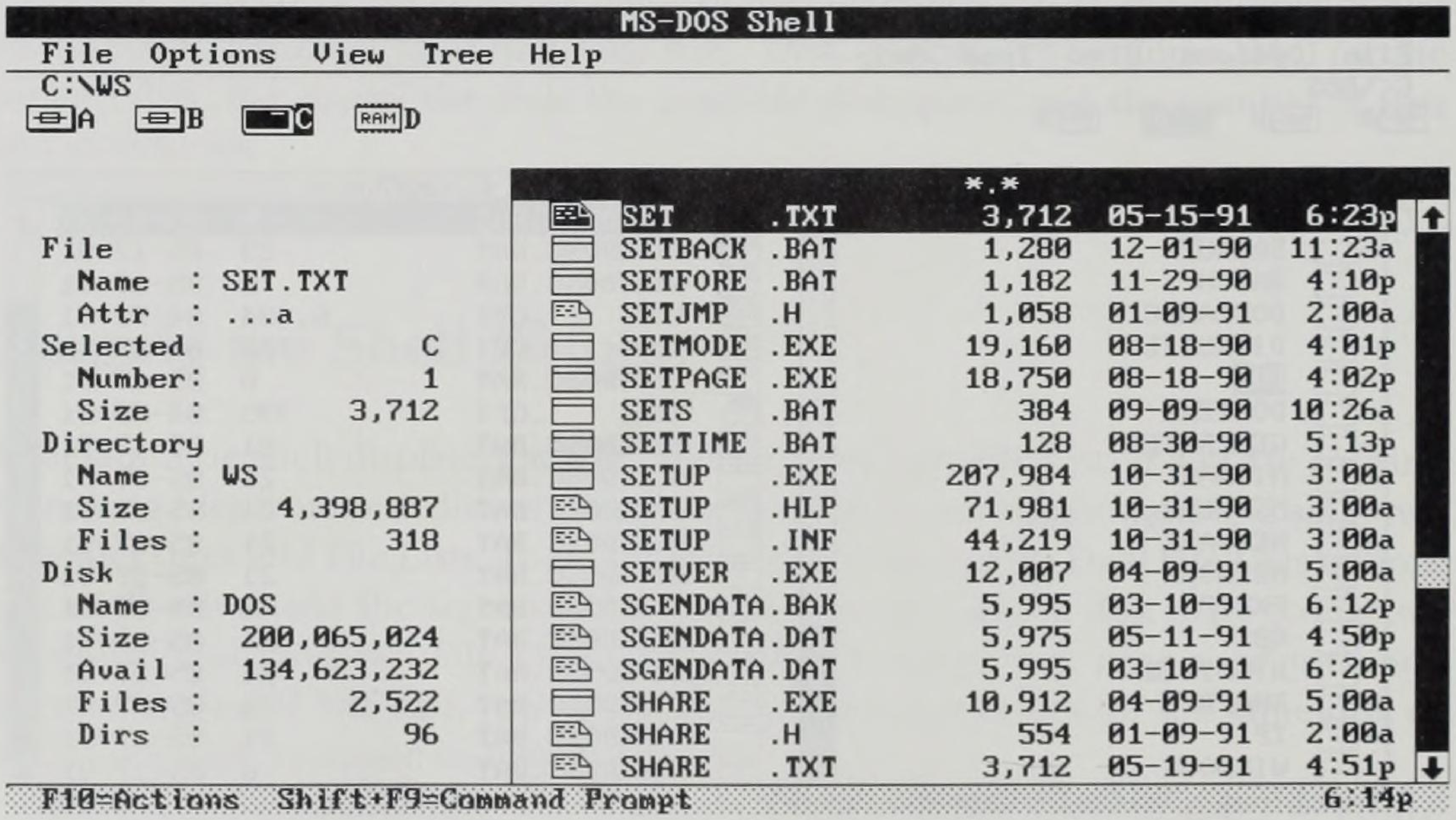
The View menu's Program List option directs the shell to remove the directory tree and File List, expanding the Program List to full screen, as shown in Figure 39-9.

Last, the View menu Program/File Lists option directs the shell to restore its normal single directory tree, File List display with the Program List at the bottom of the display.

Running Programs from the Program List

The Program List is a collection of programs and groups of programs (similar to a directory) from which you can quickly execute your commonly used commands. The shell provides three program entries and the Disk Utilities program group. In Chapter 40 you will learn how to add your own programs to this group.

To execute a program from within the Program List, you can double-click on the program or use your keyboard to highlight the program, pressing ENTER to execute it. The Command Prompt entry lets you temporarily exit the shell to DOS, as


MS-DOS Shell
File Options View Tree Help
C:\WS
A
B
C
RAM D
.
F10=Actions Shift+F9=Command Prompt
6:14p

```
File                      SET      .TXT       3,712  05-15-91   6:23p
  Name   : SET.TXT        SETBACK  .BAT       1,280  12-01-90  11:23a
  Attr   : ...a           SETFORE  .BAT       1,182  11-29-90   4:10p
Selected              C   SETJMP   .H         1,058  01-09-91   2:00a
  Number:             1   SETMODE  .EXE      19,160  08-18-90   4:01p
  Size   :        3,712   SETPAGE  .EXE      18,750  08-18-90   4:02p
Directory                 SETS     .BAT         384  09-09-90  10:26a
  Name   : WS             SETTIME  .BAT         128  08-30-90   5:13p
  Size   :    4,398,887   SETUP    .EXE     207,984  10-31-90   3:00a
  Files  :          318   SETUP    .HLP      71,981  10-31-90   3:00a
Disk                      SETUP    .INF      44,219  10-31-90   3:00a
  Name   : DOS            SETVER   .EXE      12,007  04-09-91   5:00a
  Size   :  200,065,024   SGENDATA.BAK        5,995  03-10-91   6:12p
  Avail  :  134,623,232   SGENDATA.DAT        5,975  05-11-91   4:50p
  Files  :        2,522   SGENDATA.DAT        5,995  03-10-91   6:20p
  Dirs   :           96   SHARE    .EXE      10,912  04-09-91   5:00a
                          SHARE    .H           554  01-09-91   2:00a
                          SHARE    .TXT       3,712  05-19-91   4:51p
```

Figure 39-8. *Viewing all the files on your disk*

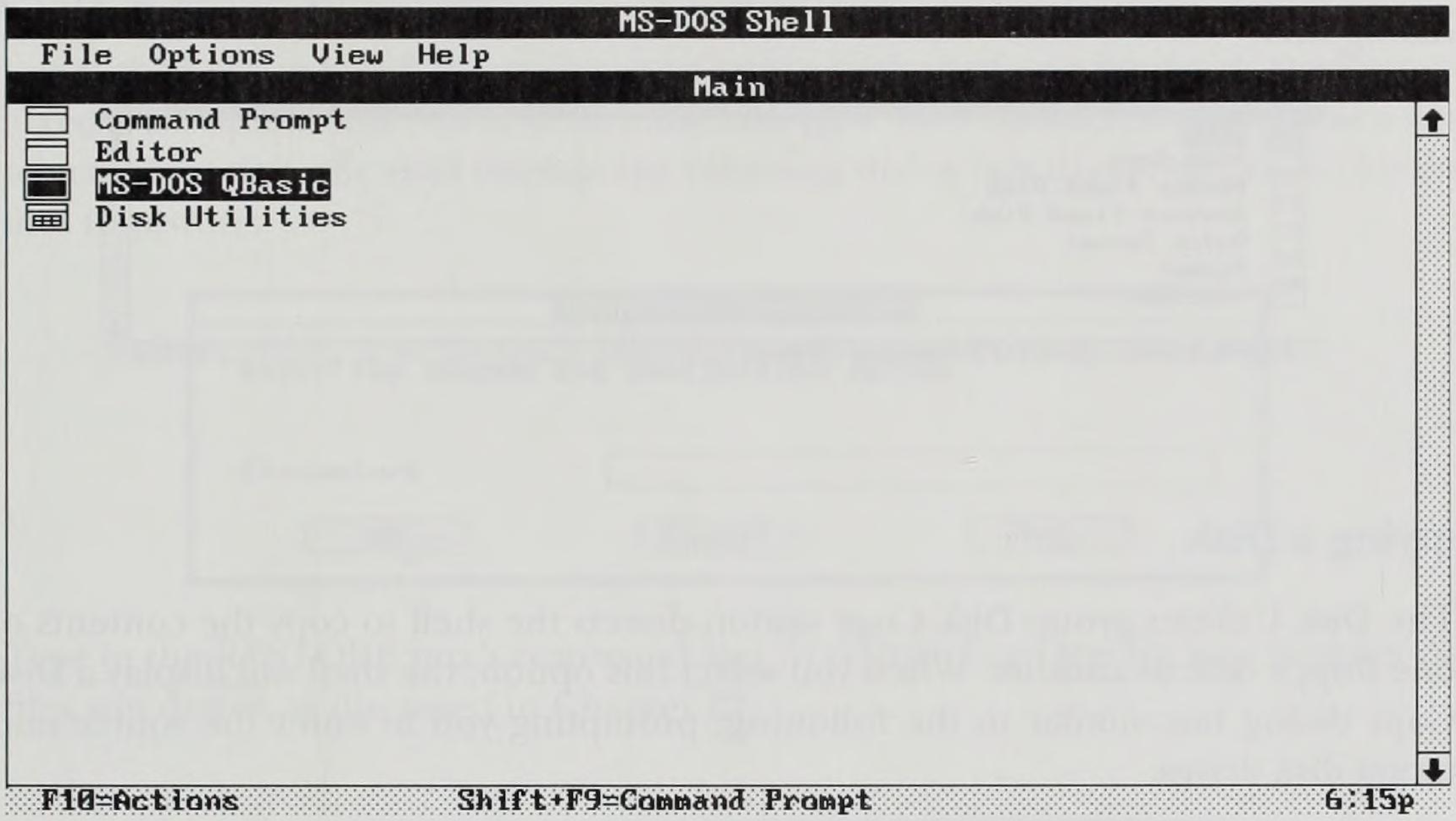

MS-DOS Shell
File Options View Help
Main
Command Prompt
Editor
MS-DOS QBasic
Disk Utilities
F10=Actions
Shift+F9=Command Prompt
6:15p

Figure 39-9. *Full-screen Program List display*

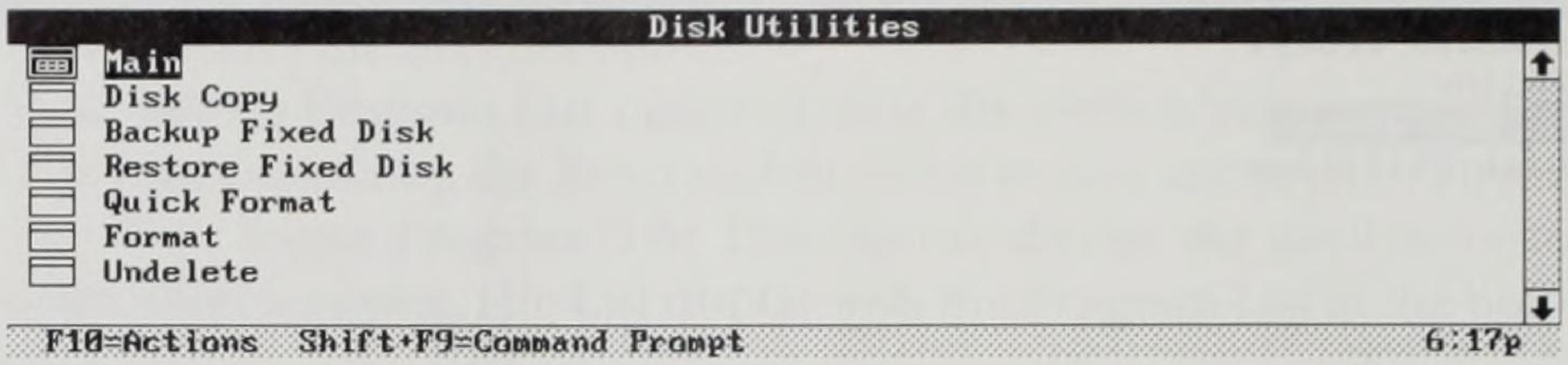
previously discussed. To return to the shell, you must issue the EXIT command. The Editor option invokes EDIT, the DOS full-screen editor discussed in Chapter 37. Likewise, the MS-DOS QBASIC option invokes Microsoft's new QBASIC interpreter, which lets you create and run your own programs written in BASIC. For more information on QBASIC, turn to the Osborne/McGraw-Hill book *QBASIC Made Easy*, 1991, by Don Inman and Bob Albrecht.

The Disk Utilities group is the most interesting menu option. You can distinguish a group entry from a program entry within the group using the icons shown in Table 39-4.

Icon	Entry Type
▭	Program Entry
▤	Program Group

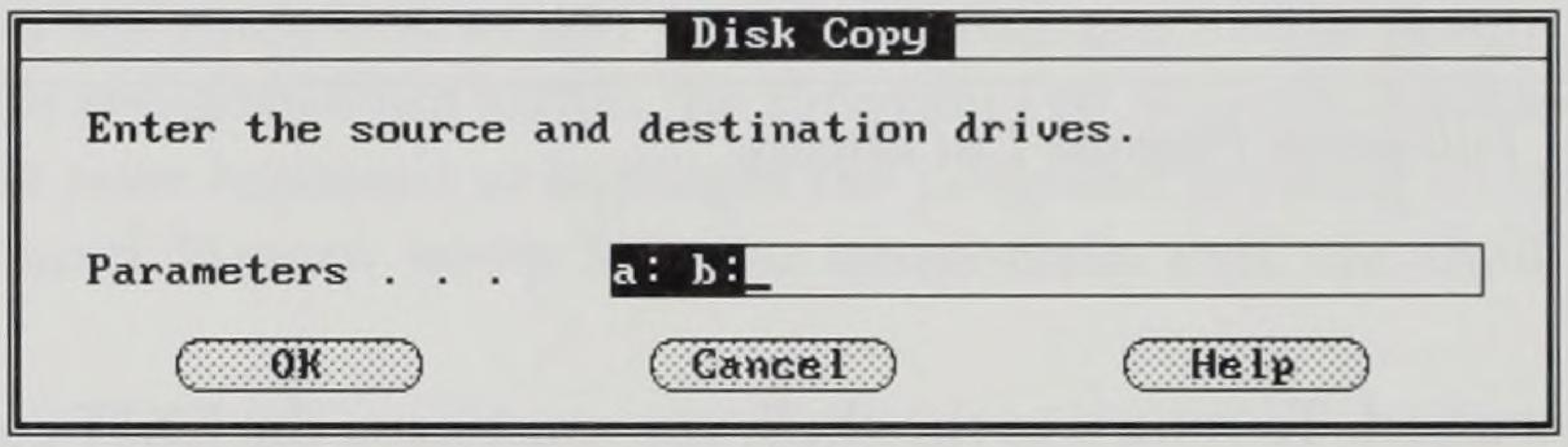
Table 39-4. *The Program Icons*

When you select the Disk Utilities group, the shell will display the following dialog box:


Disk Utilities
Main
Disk Copy
Backup Fixed Disk
Restore Fixed Disk
Quick Format
Format
Undelete
F10=Actions Shift+F9=Command Prompt
6:17p

Copying a Disk

The Disk Utilities group Disk Copy option directs the shell to copy the contents of one floppy disk to another. When you select this option, the shell will display a Disk Copy dialog box similar to the following, prompting you to enter the source and target disk drives:


Disk Copy

Enter the source and destination drives.

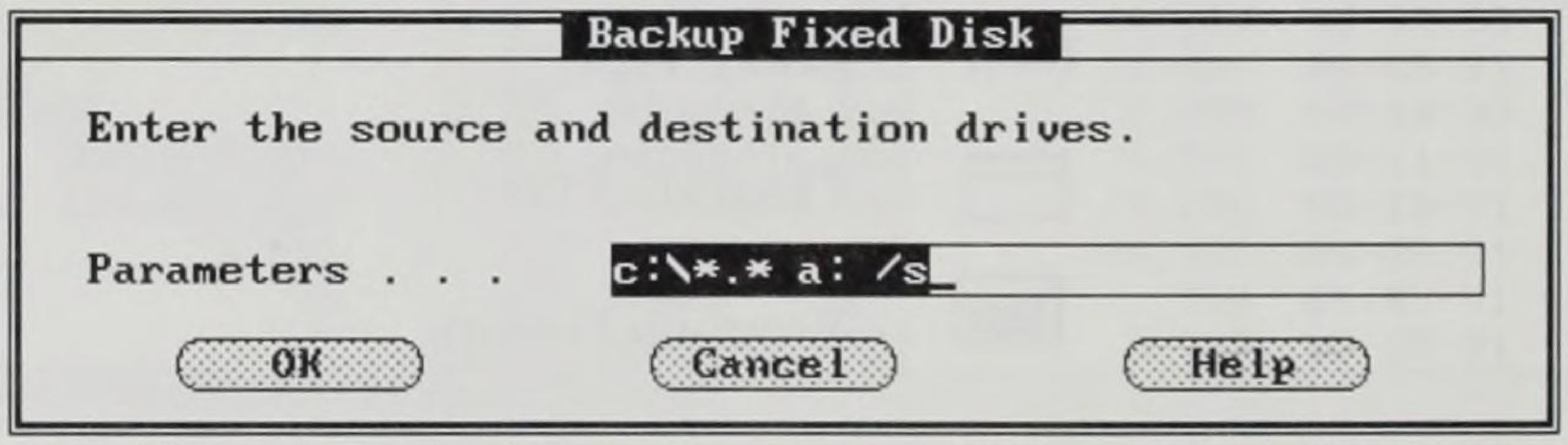
Parameters . . . a: b:_

OK
Cancel
Help

If your system has two matching floppy drives, press ENTER to select the drive A and B defaults. If you need to perform a single drive operation, type in the same source and target drive letters (such as **A: A:**). When you press ENTER or click on the OK option, the shell will invoke the DISKCOPY command, as discussed in Chapter 12.

Backing Up a Disk

The Disk Utilities Backup Fixed Disk option performs a complete backup of your hard disk, as discussed in Chapter 22. When you select this option, the shell will display the following dialog box, displaying the BACKUP command line:


Backup Fixed Disk

Enter the source and destination drives.

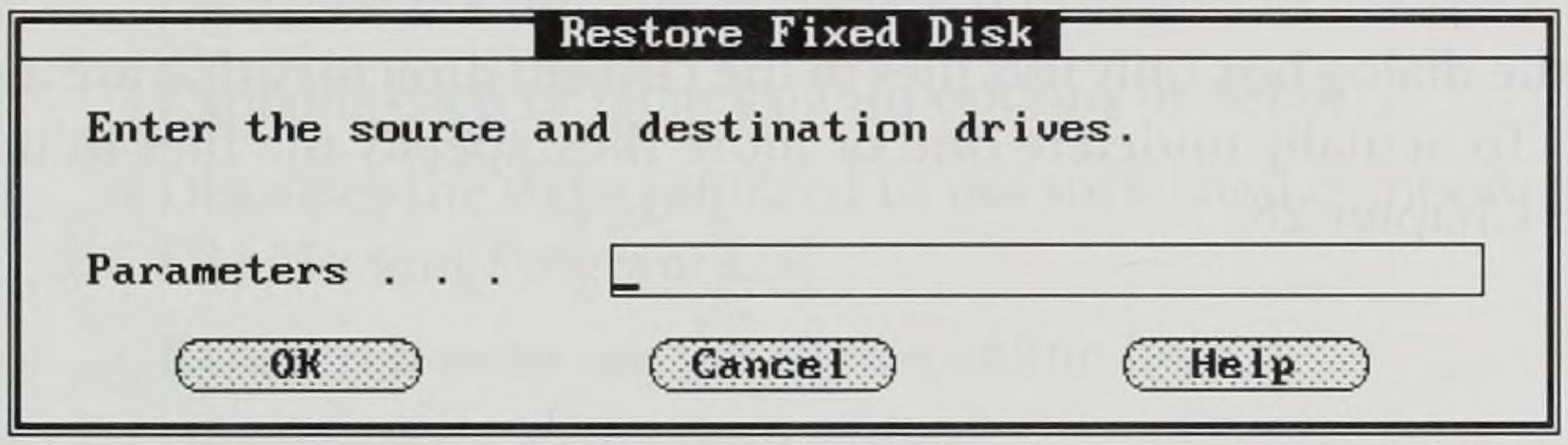
Parameters . . . c:\*.* a: /s_

OK
Cancel
Help

To back up your entire disk to floppies in drive A, press ENTER or click on the OK option. To perform an incremental backup operation or to select a different set of files, change the command line, as discussed in Chapter 22.

Restoring Fixed Disk Files

Just as the Backup Fixed Disk option performs a backup of your hard disk, the Restore Fixed Disk option lets you restore hard disk files from backup floppies. When you select this option, the shell displays the following dialog box that prompts you for the files to restore:



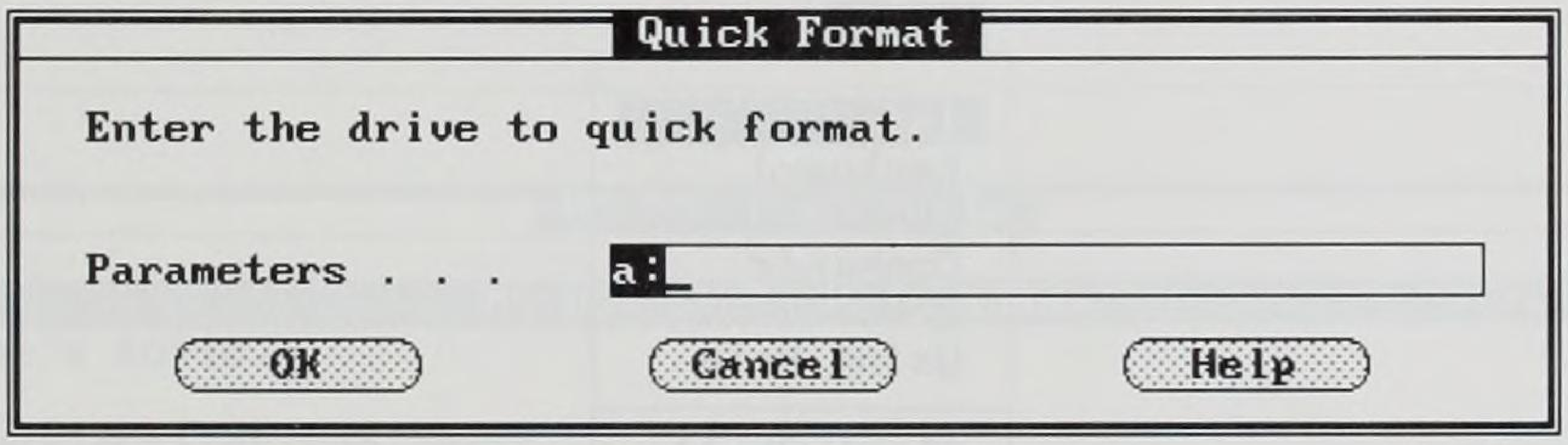
Enter the source and destination drives.

Parameters . . .

Type in the RESTORE box's command line to restore from the backup floppies the files you desire, as discussed in Chapter 22.

Formatting a Disk

The Disk Utilities Quick Format and Format options let you format a disk for use by DOS. When you select the Quick Format option, the shell displays the following Quick Format dialog box that prompts you to enter the drive to quick format:



Enter the drive to quick format.

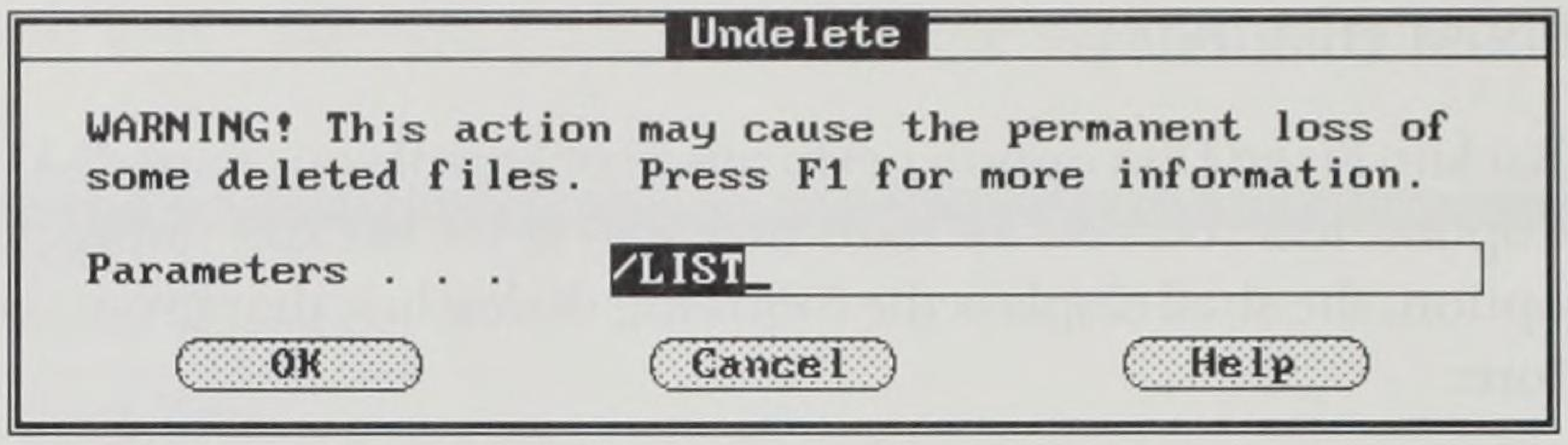
Parameters . . . a:

To format a disk other than drive A, change the drive letter specified. If you want to create a bootable disk, add the /S switch. If you need to format a smaller capacity disk, include the /4 or /F switches.

The Format option displays a similar dialog box. The difference between the two options is Quick Format performs a quick disk format, only rebuilding the file allocation tables and root directory, whereas the Format option examines the disk for damaged locations. To perform a quick format, the disk must have been previously formatted. For more information on quick format refer to Chapter 7 or to the Command Reference.

Undeleting Files

The Disk Utility Undelete command lets you undelete one or more deleted files. When you select this option, the shell will display this dialog box.



WARNING! This action may cause the permanent loss of some deleted files. Press F1 for more information.

Parameters . . . /LIST

By default, the dialog box only lists files in the current directory that are available for undeletion. To actually undelete one or more files, specify the files to undelete, as discussed in Chapter 28.

Using the Shell's Online Help

The shell provides an extensive built-in online help system that explains the different menu commands, dialog boxes, and shell components. The shell provides several ways to invoke its online help. First, if you highlight a menu option or a section of the shell itself, such as a disk drive icon, you can press F1 to scroll through a dialog box of text about the topic. Second, many dialog boxes provide built-in help. If you don't understand how you should respond to a dialog box, press F1. Last, the Help menu lets you access help on several specific topics, as shown here:

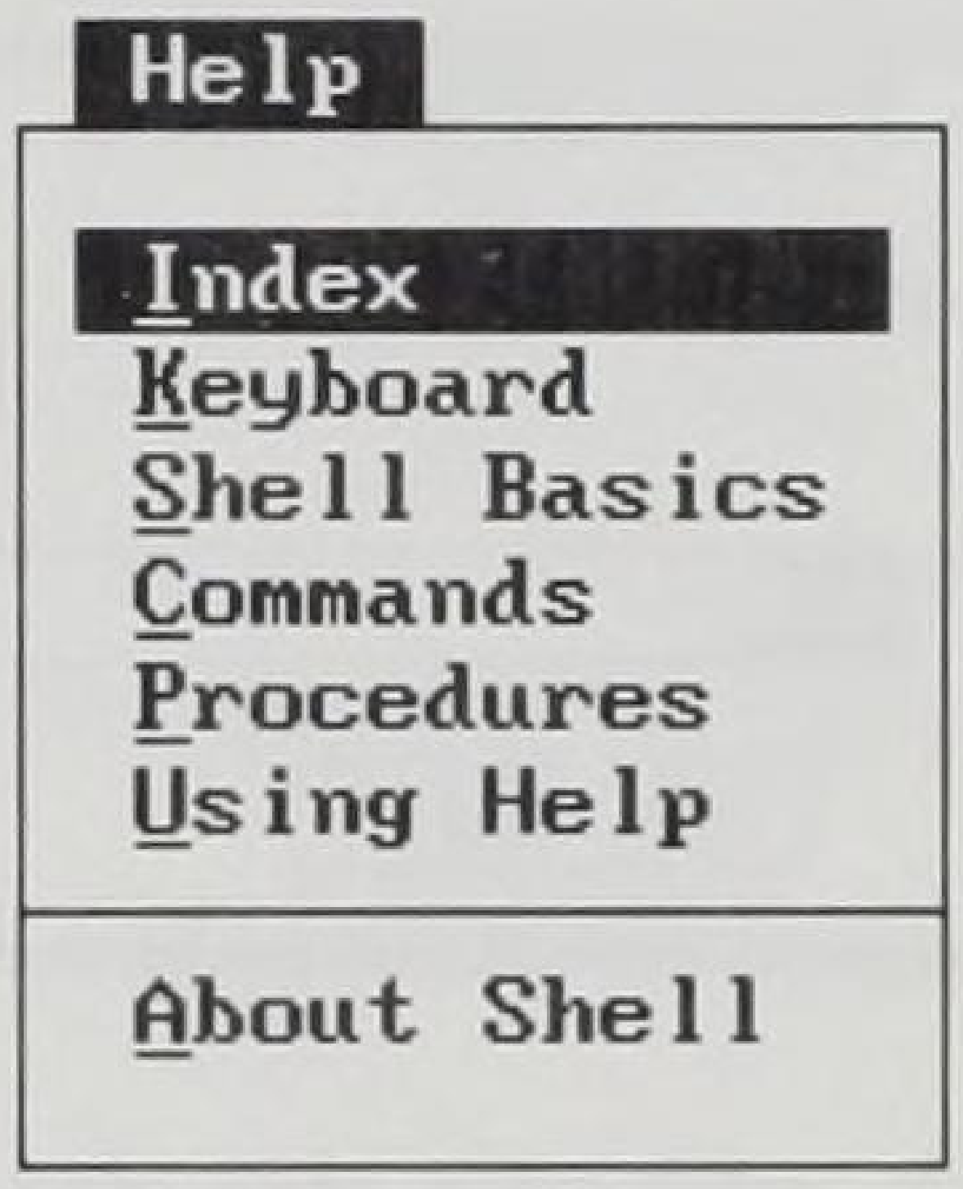


Table 39-5 briefly describes the Help menu options.

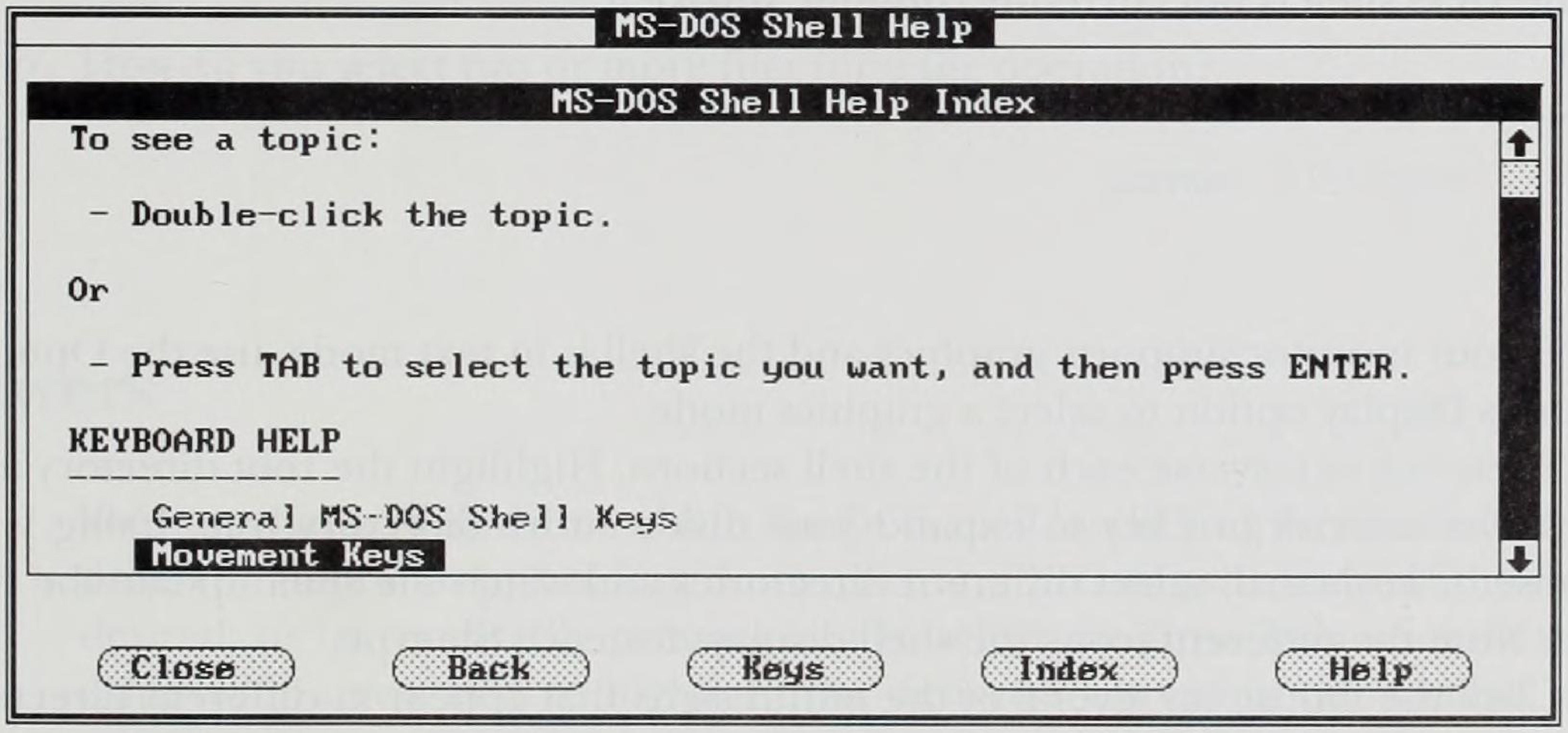
When invoked, the shell's online help displays a help window similar to the one shown in Figure 39-10.

Using your keyboard arrow keys or dragging the scroll bar slide, you can view the information the window contains. As you scroll through the text, you will notice that the window displays some text in reverse video or a different color. Such highlighted text indicates additional discussion about the topic. To view the additional text, double-click your mouse on the highlighted text or press TAB until the window highlights the text and press ENTER.

Option	Help Text
Index	Displays Help's main index letting you scroll through all the available help topics
Keyboard	Displays a list of predefined keys within the shell
Shell Basics	Discusses the fundamentals you must know to use the shell
Commands	Explains each of Help's menu options in detail
Procedures	Discusses the steps required to use such shell components as the File List and Program List
Using Help	Explains how to use the shell's online help
About Help	Displays Help's copyright notice

Table 39-5. *The Help Menu Options*

The Help window contains several options. The Close option ends the help session and removes the window. The Back button displays the previous help topic. The Keys option displays a list of related topics. The Index option redisplays Help's Index of topics, and Help displays help about the current dialog box.



To see a topic:

- Double-click the topic.

Or

- Press TAB to select the topic you want, and then press ENTER.

KEYBOARD HELP

General MS-DOS Shell Keys
Movement Keys

Figure 39-10. *The Online Help window*

SUMMARY

The DOS 5 and 6 Shell

The DOS shell is a graphical interface that lets you view and select commands and programs (keyboard or mouse), rather than typing them from the command line. Commands are logically grouped in menus. Executable commands and programs, as well as files, are also visible as icons with labels that you can choose. For many this is much easier than having to retype commands. Files can be associated with certain programs, so that choosing a file with the TXT extension will trigger the DOS editor EDIT, for example. You can view one full-page File List, two File Lists, a File List and a Program List, or a full-page Program List. You can perform DOS commands from the shell or exit (temporarily or completely) to the DOS prompt.

The DOS shell has the capacity to perform task swapping, which means that you can run a word processor, switch over to the spreadsheet that you are running, and switch back to the word processor without having to save documents and quit or restart either application (you can have more than two active tasks).

Hands On

If the DOS shell is not currently running, invoke it.

```
C:\> DOSSHELL <ENTER>
```

If your monitor supports graphics and the shell is in text mode, use the Options menu's Display option to select a graphics mode.

Press TAB to traverse each of the shell sections. Highlight the root directory and press the asterisk hot key to expand your disk's entire directory tree. Using your mouse or keyboard, select different directories and watch the shell update the File List. Note the different icons the shell displays for each file type.

Click the mouse on several of the minus signs that appear in different directory icons to collapse those directory branches.

Select the root directory. Highlight the CONFIG.SYS file and use the File menu's View File Contents option to display the file's contents.

Next, select the View menu's Dual File Lists option to display two directory trees and File Lists at the same time. Select the root directory as the current directory in the top tree and the DOS directory as the current directory in the bottom tree. Select the View menu's All Files option to display an alphabetical list of every file on your disk. Use your arrow keys or mouse to scroll through the list. Use the View menu's Program/File List option to restore the shell's standard display.

Press TAB to select the Program List and press ENTER to invoke the Command Prompt option (or click on the option with your mouse). The shell will let you temporarily exit to the DOS prompt. Invoke DIR. When the command completes, use EXIT to return to the shell.

```
C:\> EXIT <ENTER>
```

Last, invoke the Help menu and choose the Index option. Take a few minutes to scroll through the shell's online help text. Press ESC or click on the Close option to exit Help. Press ALT-F4 to exit the shell, returning to the DOS prompt.

Review

1. How do you change disk drives within the DOS shell?
2. What does it mean to expand or collapse a directory branch?
3. How do you traverse from one shell section to another?
4. List two ways to copy files using the shell.
5. How do you select two or more files for a file operation?

Answers

1. You can change disk drives within the DOS shell by clicking on the icon of the drive you desire; or by holding down CTRL and pressing the letter of the drive desired; or by repeatedly pressing TAB to select the drive icon section, then using your arrow keys to highlight the desired icon within the section, and pressing ENTER to select the drive.
2. By default, the shell only displays one level of the directory tree. When you expand a directory branch, you display the next level of a directory's tree. Likewise, when you collapse a directory branch, you remove the branch from view.
3. You can click on an item within the section you desire with your mouse, or you can repeatedly press the TAB key until the shell highlights the section.

4. To copy files using the shell, select the files with your mouse and keyboard and use the File menu's Copy option or select the files you desire and drag the files with your mouse onto the directory icon to which you want the files copied, releasing the mouse button to complete the copy operation.

5. To select two or more consecutive files with your mouse, click on the name of the first file in the list. Hold down SHIFT and click on the name of the last file in the list.

   To select two or more consecutive files with your keyboard, use the arrow keys to highlight the first file in the group. Hold down SHIFT and use the arrow keys to highlight remaining files in the group.

   To select two or more files that aren't consecutive, hold down CTRL while you click on each file. To select the files with your keyboard, use the arrow keys to select the first file you desire. Press SHIFT-F8. The shell will display the word "Add" in the status bar next to the current time. Use the arrow keys to highlight the second file desired. Press SPACEBAR to select the file. Repeat these steps to highlight and select the desired files. After all files are selected, press SHIFT-F8 again.

   To select all the files in the current directory, use the File menu Select All option.

Advanced Concepts

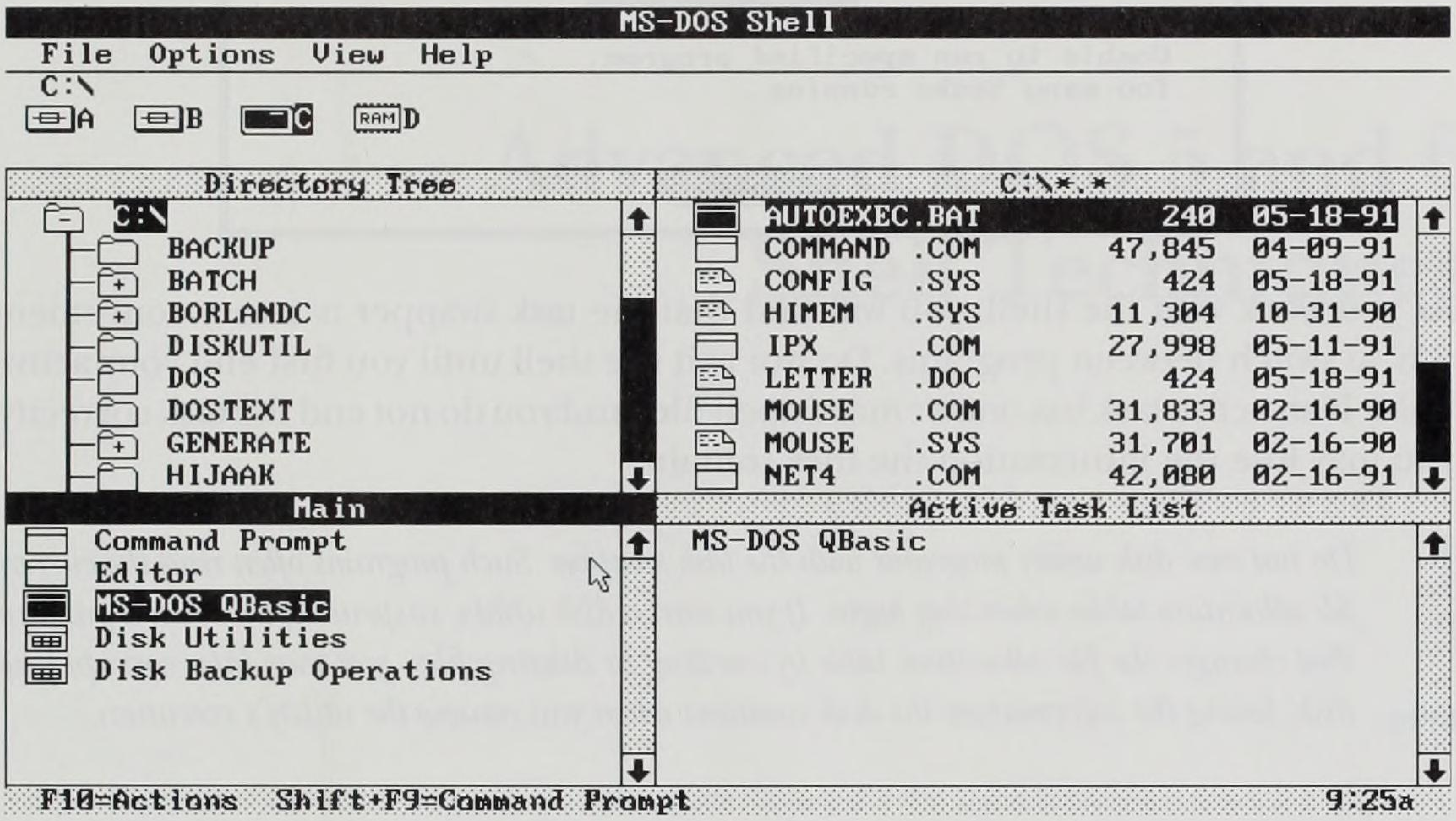
As you have learned, DOS only lets you run one program at a time with the DOS 5 shell; however, you can start one program, such as your word processor, open a file, and temporarily suspend your word processor so you can run a second program, leaving the word processing session right where you left off. Using this technique, you can start several programs, switching between the programs as required. The DOS 5 shell refers to each program as a *task,* and the shell's *task swapper* lets you quickly switch between loaded programs. The number of programs you can load depends on their size and your available memory.

To enable task swapping, select the Options menu and choose the Enable Task Swapper option. The shell will display a diamond next to the option when task swapping is enabled.

Next, run your program. Press CTRL-ESC to return to the DOS shell. The shell will divide the Program List group in half, displaying an Active Task List. If you invoke several programs, your Active Task List may appear similar to Figure 39-11.

To select a program from the Active Task List, click on the program with your mouse or press the TAB key to highlight the list, use the arrow keys to highlight the desired program, and press ENTER.

To cycle through the active programs, press ALT-TAB. The shell will cycle through the active tasks, displaying each program's name in a title bar. To select a specific program, press ENTER.



```
                                MS-DOS Shell
 File  Options  View  Help
 C:\
 A   B   C   D

            Directory Tree                                C:\*.*
 C:\                                   AUTOEXEC.BAT        240  05-18-91
  ├BACKUP                              COMMAND .COM     47,845  04-09-91
  ├BATCH                               CONFIG  .SYS        424  05-18-91
  ├BORLANDC                            HIMEM   .SYS     11,304  10-31-90
  ├DISKUTIL                            IPX     .COM     27,998  05-11-91
  ├DOS                                 LETTER  .DOC        424  05-18-91
  ├DOSTEXT                             MOUSE   .COM     31,833  02-16-90
  ├GENERATE                            MOUSE   .SYS     31,701  02-16-90
  ├HIJAAK                              NET4    .COM     42,080  02-16-91
                  Main                          Active Task List
 Command Prompt                        MS-DOS QBasic
 Editor
 MS-DOS QBasic
 Disk Utilities
 Disk Backup Operations

 F10=Actions  Shift+F9=Command Prompt                                  9:25a
```

Figure 39-11. *The Active Task List*

To end an active task, select the program and end the program as you normally would. If a program hangs, select the Task List, highlight the program, and press DEL.

If you are using an EGA monitor and the shell does not correctly update your screen when swapping tasks, install the EGA.SYS device driver in your CONFIG.SYS file.

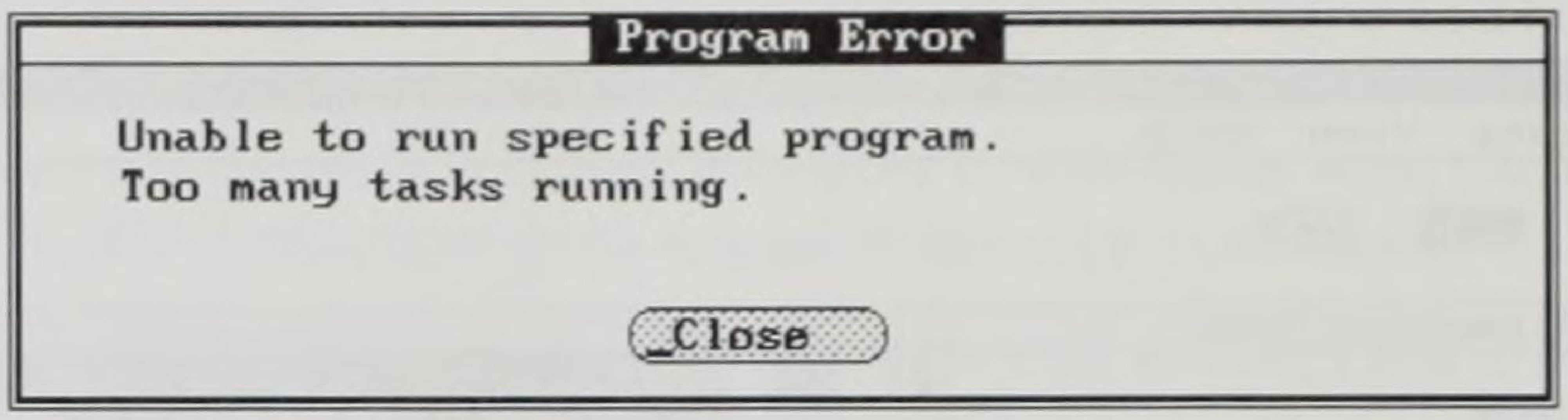
NOTE

To add a program appearing in the File List to the Active Task List, hold down SHIFT and double-click on the program or use your arrow keys to highlight the program and press SHIFT-ENTER. The shell will not run the program; rather, it will add the task to the Active Task List for selection.

The task swapper does not perform multitasking—it does not run more than one program at a time. Instead, the task swapper lets you run one program at a time, remembering where you left off in the other programs so it can later resume their execution.

NOTE

If you attempt to load more tasks than your memory can hold, the shell will display the following Program Error dialog box, indicating that it is unable to load the task.



```
Unable to run specified program.
Too many tasks running.
```

As you work with the shell, you will find that the task swapper is a very convenient way to switch between programs. Do not exit the shell until you first end your active tasks. If an active task has one or more open files and you do not end the task correctly, you may lose the information the files contain.

WARNING *Do not run disk utility programs with the task swapper. Such programs often read the current file allocation tables when they begin. If you start a disk utility, suspend it, and run a program that changes the file allocation table by creating or deleting files, you may later corrupt your disk, losing the information the disk contains when you resume the utility's execution.*

Key Terms

Collapse Make subdirectories that were displayed invisible.

Expand Make subdirectories visible in a File List.

File List The DOS 5 and 6 shell's display of files in the current directory.

Program List The DOS 5 and 6 shell's display of executable programs.

Task swapper The program within the DOS 5 and 6 shell that lets you temporarily suspend one program so you can run another, letting you resume the original program's execution at a later time at the same point where you left off.

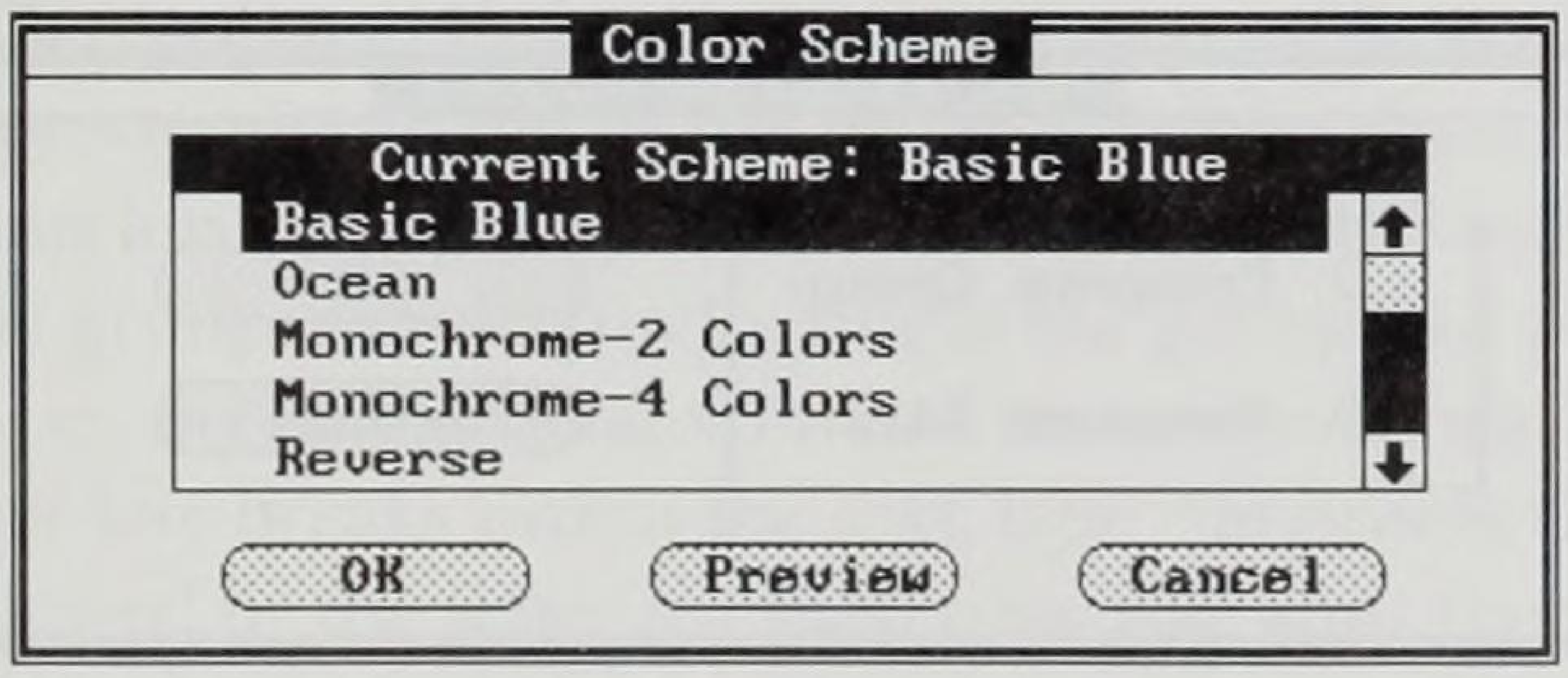
Chapter 40

Advanced DOS 5 and 6 Shell Techniques

In Chapter 39 you learned how to use the DOS 5 and 6 shell to perform file-manipulation commands such as Copy, Move, Rename, and Delete. You also learned how to quickly traverse your directory tree or two different directory trees. In this chapter you will learn how to customize the DOS 5 and 6 shell by selecting screen colors and adding programs or program groups to the Program List. In the "Advanced Concepts" section, you will learn how to build custom dialog boxes that the shell displays to prompt the user for command-line parameters when a specific program entry is selected.

Setting Your Screen Colors

The shell provides eight different color settings you can select using the Options menu Colors option. When you select this option, the shell will display this Color Scheme dialog box:



Color Scheme
Current Scheme: Basic Blue
Basic Blue
Ocean
Monochrome-2 Colors
Monochrome-4 Colors
Reverse
OK
Preview
Cancel

Using your arrow keys or mouse, highlight the color desired. If you want to see how the color will look before you select it, choose the Preview option. To select the color, choose the OK option. If you decide to leave the existing color unchanged, press ESC or select the Cancel option.

Working with Program Groups

A *program group* is a collection of related programs, such as a word processor, spell checker, and thesaurus. If you have a program you use on a regular basis, you may simply want to add the program entry to the Main Group itself. However, if you have a set of related programs, you'll want to create a separate program group to organize your programs, much as you would store related files in a directory.

You might, for example, create a group named Disk Backup Operations that contains the command for a monthly backup and an incremental backup, as discussed in Chapter 22. To add the group, you would first select the shell's Program List. Next, invoke the File menu. When the Program List is selected, the options in the File menu change, as shown here:

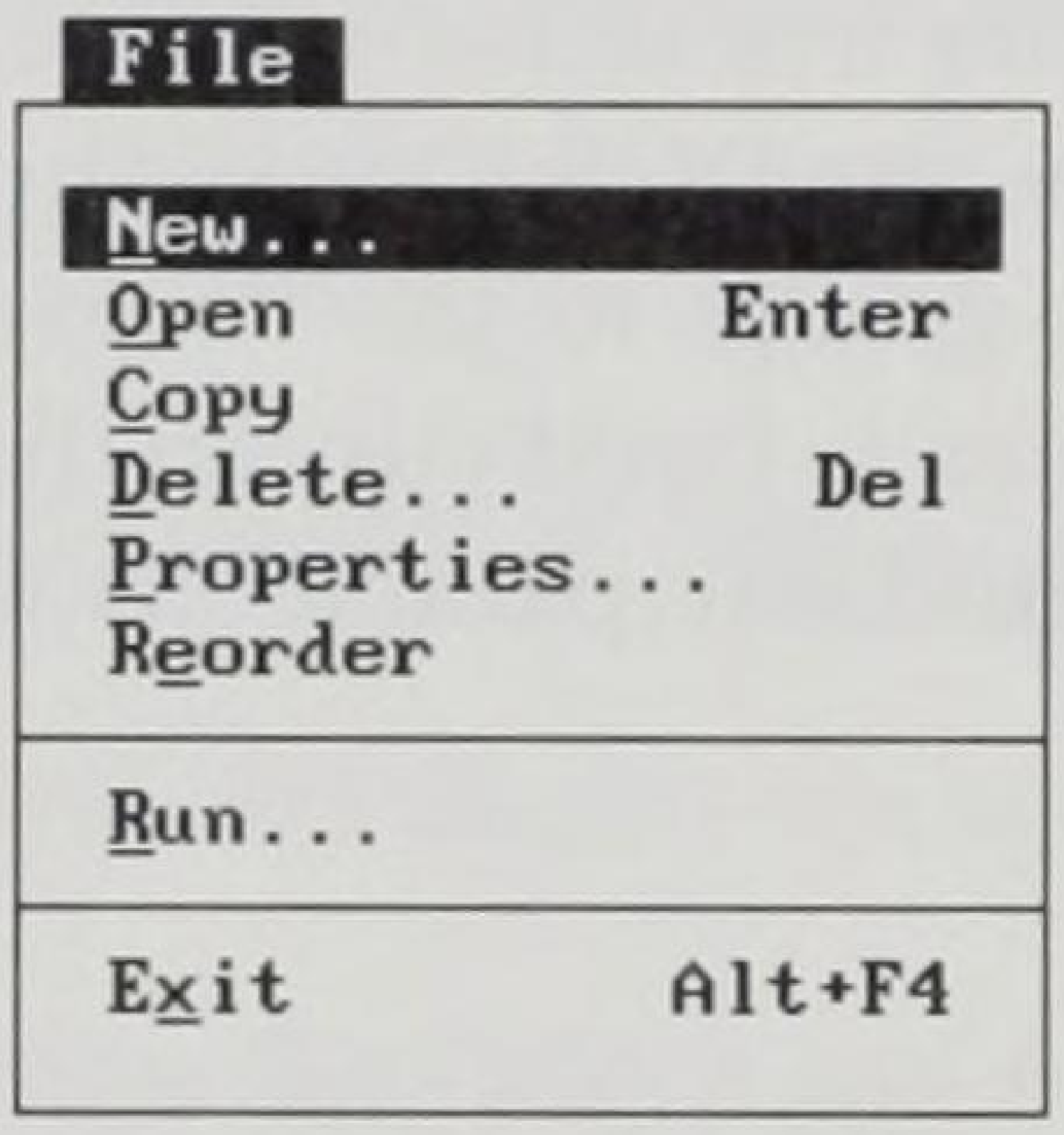
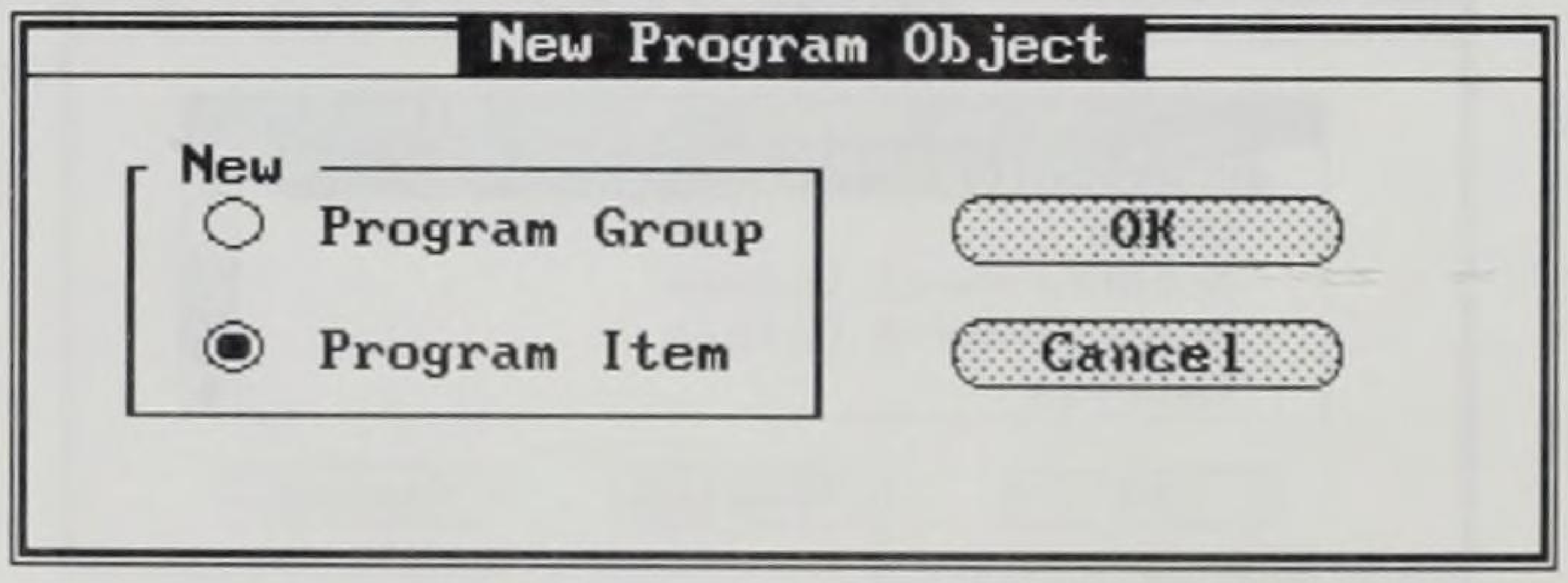


Table 40-1 briefly describes each menu option.

To add a program group, select the File menu and choose the New option. The shell will display the New Program Object dialog box, asking you if you want to add a program entry or program group.

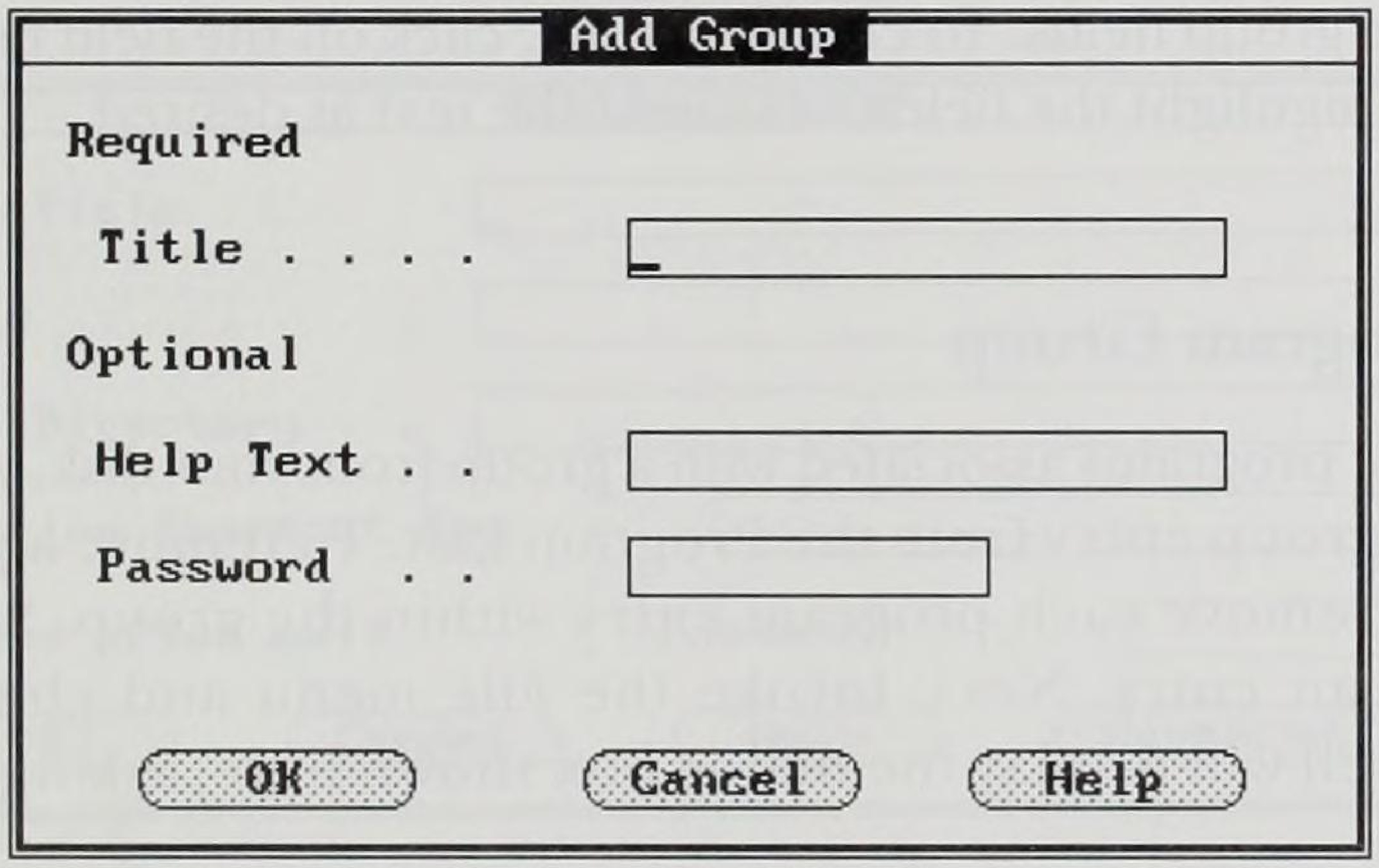


New
○ Program Group
● Program Item

Option	Function
New	Adds a new program or group entry to the current program group
Open	Displays the contents of the selected program group
Copy	Copies a program entry into a specific program group
Delete	Removes an empty program group from within the current group
Properties	Specifies the characteristics of a program or program group
Reorder	Moves a program or group entry within the Program Group menu
Run	Lets you run a command by entering the command name and its optional parameters
Exit	Quits the DOS shell

Table 40-1. *Shell File Menu Options When the Program List Is Selected*

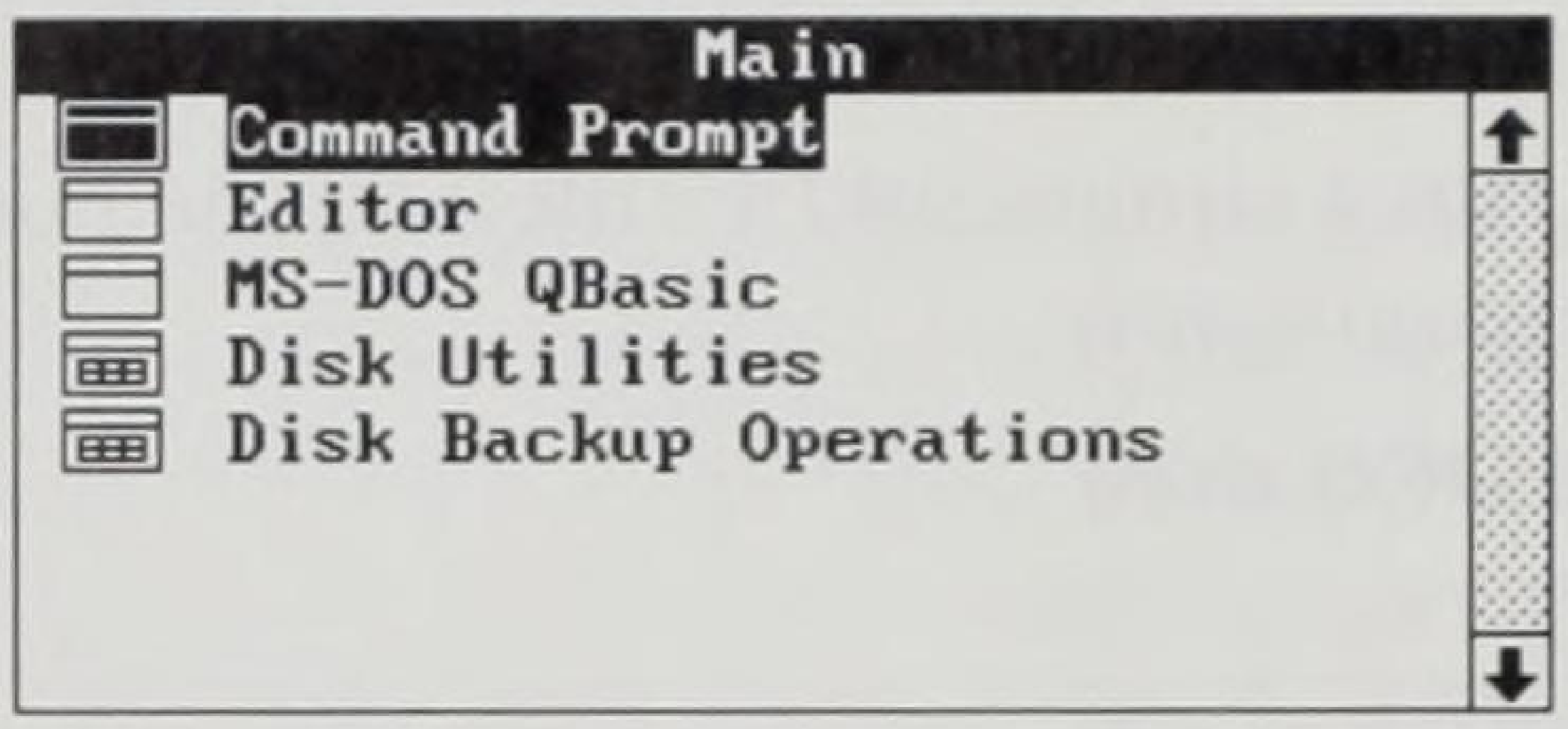
To add a program group, select the Program Group option and press ENTER or use your mouse to click on OK. The shell will display the Add Group dialog box, prompting you to type in the group specifics.



The Title option lets you specify a title containing up to 23 characters, which later appears within the group menu and as the group title bar when the group is open. The Help Text option lets you specify up to 255 characters of online help the shell will display about the group, should you highlight the group and press F1. If you type in the text as sentences, the shell will later format the text to fit within the dialog box. If you want to control line breaks within the text, type ^m (insert the caret character (^) immediately followed by the letter m) each time you want the shell to perform a carriage return and linefeed operation.

The Password option lets you specify a password up to 20 characters long that the user must type to access the group. Do not rely on the password as a security measure. As you learned in Chapter 39, pressing ALT-F4 lets the user exit the shell, at which point the user can issue the commands from the DOS prompt. In general, the password is only useful to prevent another user from adding or deleting program entries within the group.

After you complete the group fields, press ENTER or use your mouse to click on the OK option. The shell will add the group, as shown here:

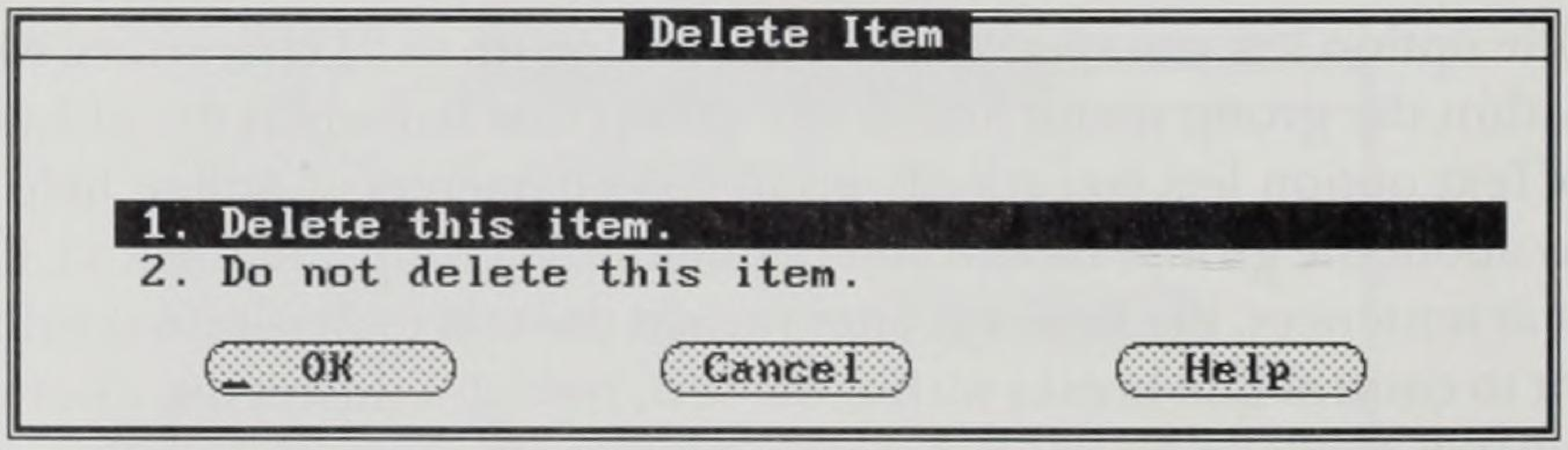


Editing an Existing Program Group

Should you later need to change a group's title, help, or password, highlight the group and select the File menu's Properties option. If the group has a password, the shell will display a dialog box asking you to enter the password. If the password you enter is correct, the shell will display a dialog box, similar to the Add Group box, that lets you edit the group fields. To change a field, click on the field using your mouse, or press TAB to highlight the field. Next, edit the text as desired.

Removing a Program Group

If you delete the programs associated with a group from your disk, you will also want to remove the group entry from the Program List. To remove a program group, you must first remove each program entry within the group. Select the group and highlight an entry. Next, invoke the File menu and choose the Delete option. The shell will display the dialog box shown here, asking you to confirm the deletion.



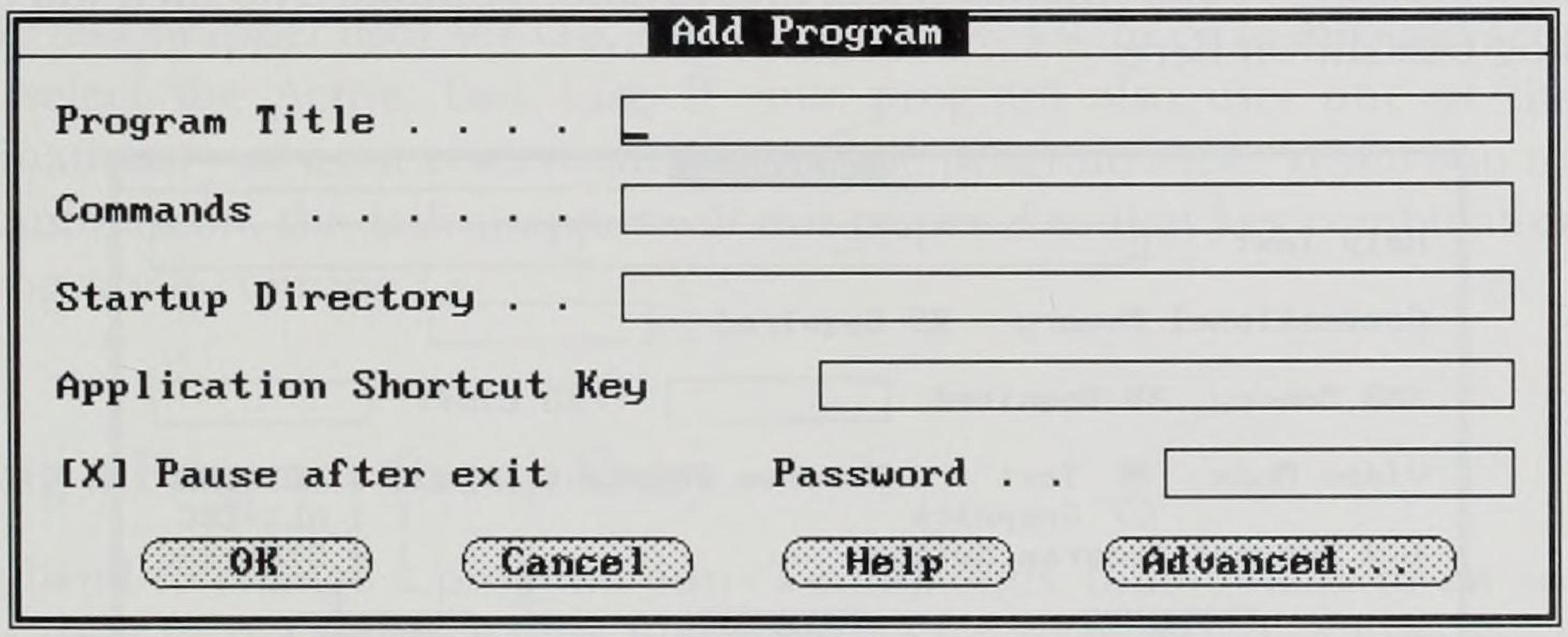
Select option 1 to delete the entry. Repeat this process for each program entry in the group. Next, press ESC to exit the program group to the next higher level. Select the File menu's Delete option. The shell will display a dialog box, similar to the Delete Item Box, asking you to confirm the deletion. Select option 1 to delete the group. If the group has a password, the shell will prompt you to enter the password before removing the group.

Changing the Order of Group Items

For convenience, you probably want to place your most commonly used program and group entries at the top of the group list. To move an entry, highlight the entry and select the File menu's Reorder option. Next, highlight the entry in the list before which you want to move the program or group and press ENTER. You must repeat this process for each entry you wish to move.

Working with Program Entries

To add a program entry to a group, select the desired group and invoke the File menu, choosing the New option. The shell will display the New Program Object dialog box, shown earlier in the chapter, asking you if you want to add a program or group item. Select the Program Item option. The shell will display the Add Program dialog box, prompting you to enter the program item information.



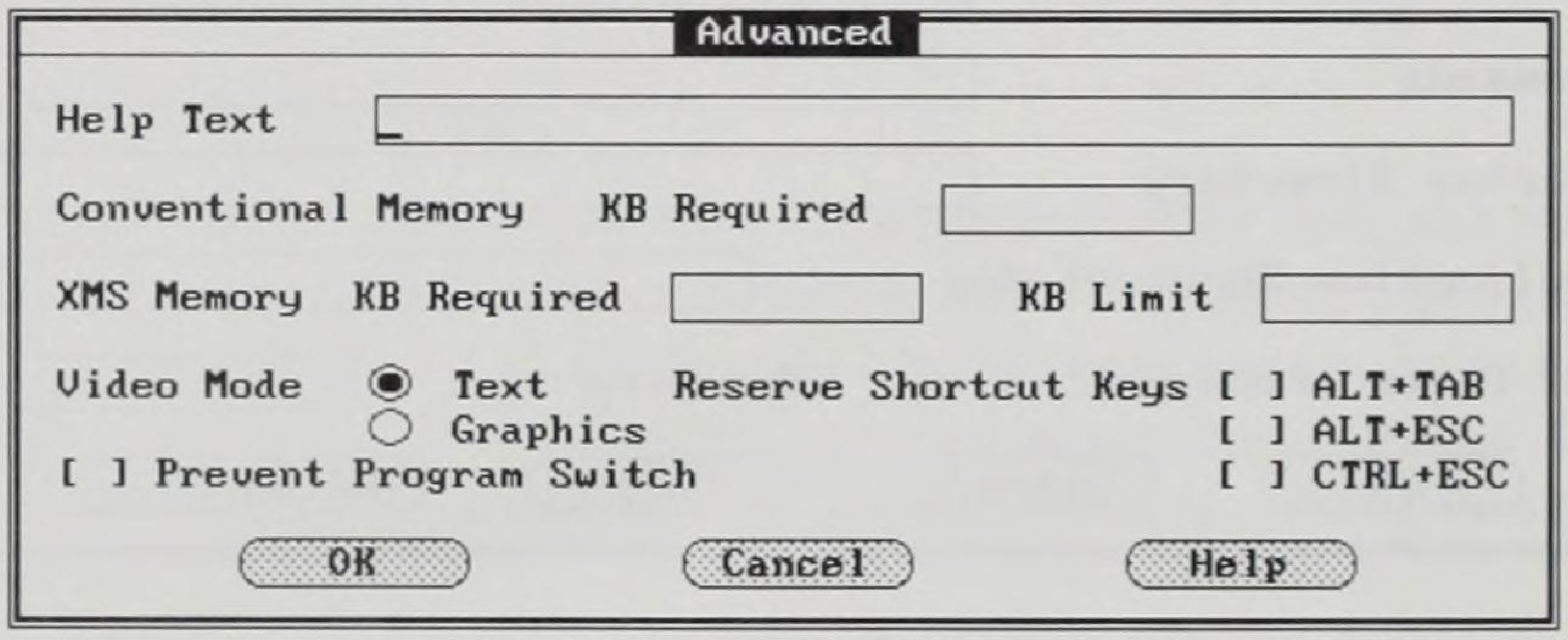
[X] Pause after exit

The Program Title option lets you specify up to 23 characters that will appear in the group menu as the entry's title. Using the backup example, you might use the title "Daily Backup." The Commands option lets you specify the command, or series of commands, the shell needs to issue when you invoke this option.

If the shell needs to perform two or more commands, such as first selecting a specific directory before running the program, separate each command with a semicolon. The commands can total up to 255 characters. If one of the commands is a DOS batch file, use the CALL command to invoke it. In the case of the incremental backup, you would enter the DOS 5 command **BACKUP C:\*.* A: /S /A /M**, as discussed in Chapter 22. If you are using DOS 6, you would enter an equivalent MSBACKUP command. In the "Advanced Concepts" section you will learn how to direct the shell to display a dialog box of options when the program runs, similar to the dialog boxes used by the commands in the Disk Utilities group discussed in Chapter 39. The Startup Directory specifies a directory name you want the shell to select as the current directory before the program runs. If you don't specify a directory, the shell uses the current directory. The Application Shortcut key lets you specify a hot key combination that directs the task swapper discussed in the "Advanced Concepts" section of Chapter 39 to automatically select the program if the program is present in the active task list. To select a hot key, hold down SHIFT, ALT, or CTRL and press a key. If you press SHIFT and F10 for example, the dialog box will display the key as "SHIFT+F10". Do not use the key combinations listed in Table 40-2 which are reserved by the shell.

The Pause after exit option directs the shell to display a message asking the user to press any key to continue when the program ends, before returning to the shell. With this option disabled, you may not be able to view the program's last screenful of output before your screen redisplays the shell. The Password option lets you specify a password up to 20 characters that the user must enter before running the program. As is the case with program group passwords, do not rely on the password for security. Instead, only use a password to prevent a user from changing one of the program entry's properties.

For most programs, the information you specify in this dialog box will be enough to successfully run the program, and you can press ENTER or click on the OK option to place the entry in the Program List. If you want to specify online help text or the program's memory requirements, select the Advanced option. The shell will display the dialog box shown here:


Advanced

Help Text

Conventional Memory KB Required

XMS Memory KB Required KB Limit

Video Mode ● Text ○ Graphics

Reserve Shortcut Keys ☐ ALT+TAB ☐ ALT+ESC ☐ CTRL+ESC

☐ Prevent Program Switch

CTRL-C	CTRL-[	SHIFT-CTRL-M	SHIFT-CTRL-[
CTRL-M	CTRL-5*	SHIFT-CTRL-I	SHIFT-CTRL-5*
CTRL-I	CTRL-H	SHIFT-CTRL-H	

Table 40-2. *Key Combinations Reserved by the DOS Shell (*indicates the numeric keypad number 5)*

The Help Text option lets you specify up to 255 characters of online help the shell will display about the program should the user highlight the program and press F1. If you type in the help text as sentences, the shell will later format the text to best fit within the help dialog box. If you want to control line breaks within the text, type ^m (insert a caret character immediately followed by the letter m) each time you want the shell to perform a carriage return and linefeed operation.

The Conventional Memory KB Required option lets you specify the minimum amount of conventional memory, in kilobytes, the program requires before the task swapper can run the program. The default value is 128Kb. The XMS Memory KB Required option lets you specify the minimum amount of extended memory the program must have before the task swapper can run it. The default value is zero. The XMS Memory KB Limit specifies the maximum amount of extended memory, in kilobytes, the program can use. The default value is 384Kb, or the amount of the extended memory if the system has less than 384Kb.

When the task swapper changes from one active task to another, the swapper must save the current task's screen display. The Video Mode option controls how much memory the swapper must reserve to store the screen display. Unless you are using a CGA monitor, select Text mode for all programs. If you are using a CGA monitor and the task swapper does not correctly restore your screen, select Graphics mode.

By default, pressing ALT-ESC or CTRL-ESC directs the task swapper to either cycle through active tasks, or to display the Active Task List. If you have a program whose execution you don't want interrupted, select the Prevent Program Switch option. When you select this option, the only way to return to the shell when the program runs is to end the program.

The task swapper uses ALT-TAB, ALT-ESC, and CTRL-ESC to cycle through active tasks or to select the Active Task List. If your program also uses one of these key combinations, you must reserve the key for the program's use. When you reserve a key combination, the task swapper will not respond to that key combination when the program is running.

Changing a Program Group Entry

If you need to change a program entry's commands, online help, or an advanced option used by the task swapper, highlight the program entry and choose the File menu's Properties option. The shell will display a dialog box similar to the Add Program box shown earlier in this chapter. Change the fields required and select OK to save your changes.

Copying a Program Entry to a New Group

If accidentally you place a program entry in the wrong group, or later want to move an entry to a different group, highlight the entry and select the File menu's Copy option. Next, display the group within which you want the item and press ENTER. The item will appear in the group's Program List. Next, select the entry's original group and remove the entry with the File menu's Delete option. The shell will display a dialog box similar to the Delete Item box shown earlier. Select option 1 to delete the entry.

SUMMARY

Screen Colors

The DOS 5 and 6 shell offers eight different color settings, available through the Colors option of the Options menu.

Program Groups

Program groups are collections of similar programs. By grouping related programs, you can use them together easily, avoid a cluttered Program List, and even offer a measure of password security to them.

When you make new groups and add programs to them, you can specify the name displayed for those groups and any hot keys to access them. You can also specify help text to be displayed about a program when a user highlights the program and presses F1, and specify how the program is to interact with the task swapper.

Hands On

Select the Program List, invoke the File menu, and choose the New option. When the shell displays a dialog box prompting you to select a group or program item, select the Program Group item. For the group title, type in **Disk Backup Operations**. Within the Help Text option, type in a brief description of an incremental and complete disk backup, as discussed in Chapter 22. Select OK to add the group. Next, select your newly created group.

Invoke the File menu and choose the New option. When the shell displays a dialog box prompting you to select a group or program item, select the Program Item option. For the program title, type **Daily Backup**. If you are using DOS 5, in the command field, type **BACKUP C:\*.* A: /S /A /M**. Select OK to add the program entry. If you are using DOS 6, type the MSBACKUP command you normally issue to back up your entire disk.

Invoke the File menu and again choose New, selecting a second program item. For the program title, type **Monthly Backup**. Again, if you are using DOS 5, type

BACKUP C:\*.* A: /S. Likewise, in DOS 6, type in an equivalent MSBACKUP command. Select the Advanced option and type in help text describing the complete disk backup. Select the OK option to remove the Advanced dialog box and OK a second time to add the program entry.

Highlight the Daily Backup option. Select the File menu and choose the Properties option. When the dialog box displays the option's current settings, select the Advanced option and add help text describing an incremental backup operation. Select OK to remove the Advanced dialog box and OK to save your changes.

You can now perform a complete disk or incremental backup operation directly from the Program List.

Review

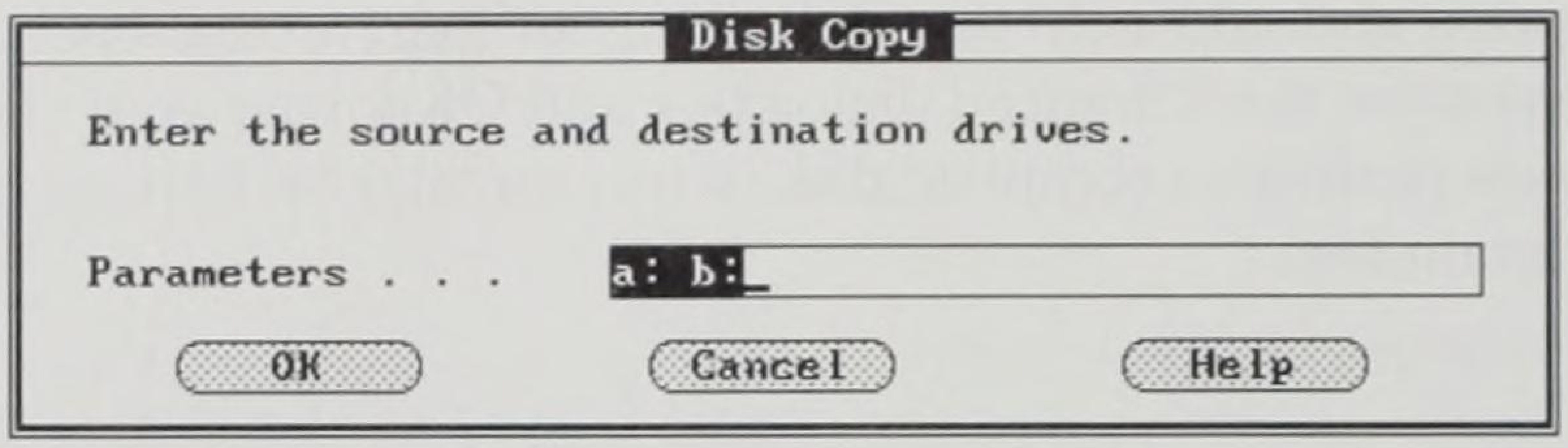
1. When should you use a program group instead of a program entry?
2. With the exception of the Help Text option, when must you specify values for the program entry options that appear in the Advanced dialog box?
3. How can you change an existing program group or program entry?

Answers

1. If you have a set of two or more related programs, you should place them within a program group. If the shell's Program List is becoming cluttered, you should organize the programs into groups, much as you would use subdirectories to organize your files.
2. The only time you need to complete options in the Advanced dialog box for a program entry is when you need to specify the program's attributes to the task swapper, and you plan to run the program while the task swapper has other programs in memory.
3. To change an existing program or group entry, highlight the entry and select the File menu Properties option. The shell will display a dialog box that lets you edit the entry's fields.

Advanced Concepts

In Chapter 39, the programs in the Disk Utilities group displayed a dialog box requesting command-line information before they ran. For example, if you selected the Disk Copy operation, the shell displayed the Disk Copy dialog box:

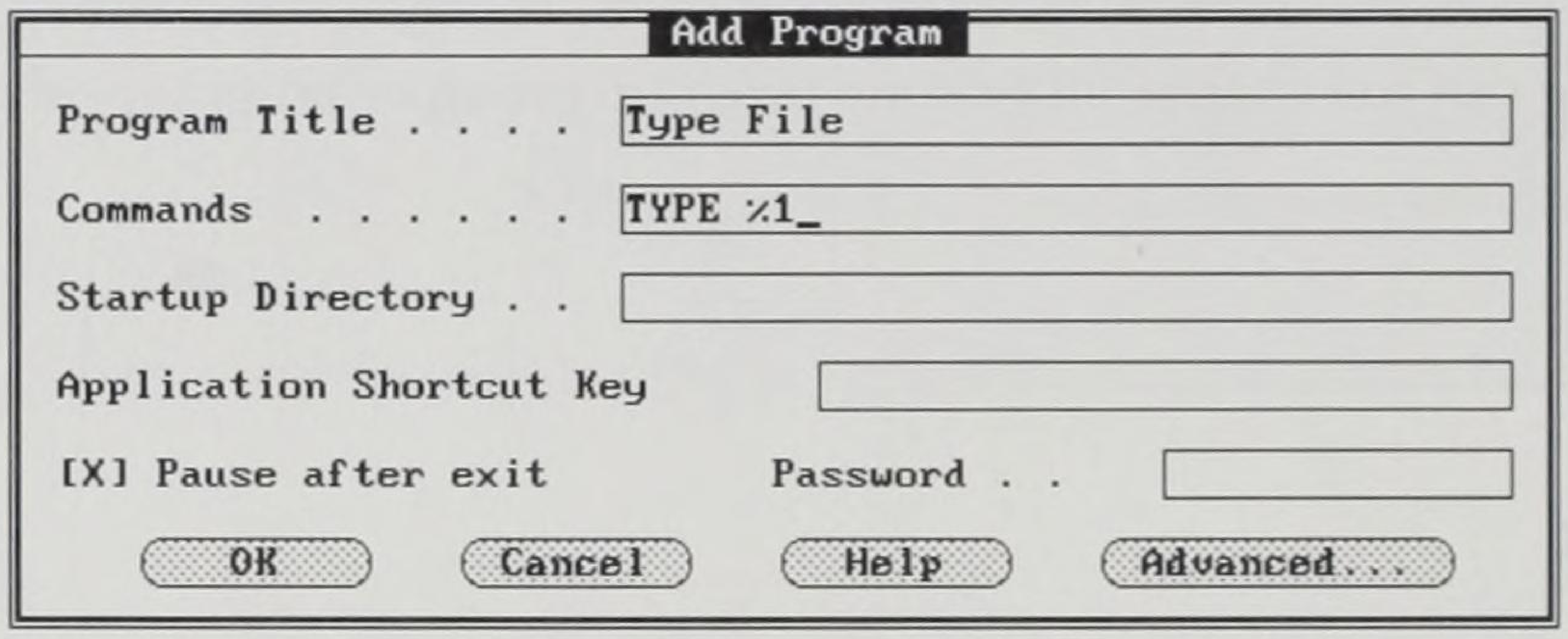


Disk Copy

Enter the source and destination drives.

Parameters . . . a: b:_

When you add your own program entries, the shell lets you design your own dialog boxes. To create a custom dialog box, you must reference a replaceable parameter %1 through %9 within the program entry's command line. For example, assume you are creating a program entry named Type File that uses the TYPE command to display the contents of the file the user specifies. Within the program's command, you would place the command TYPE %1, as shown here:



Add Program

Program Title . . . . Type File

Commands . . . . . . TYPE %1_

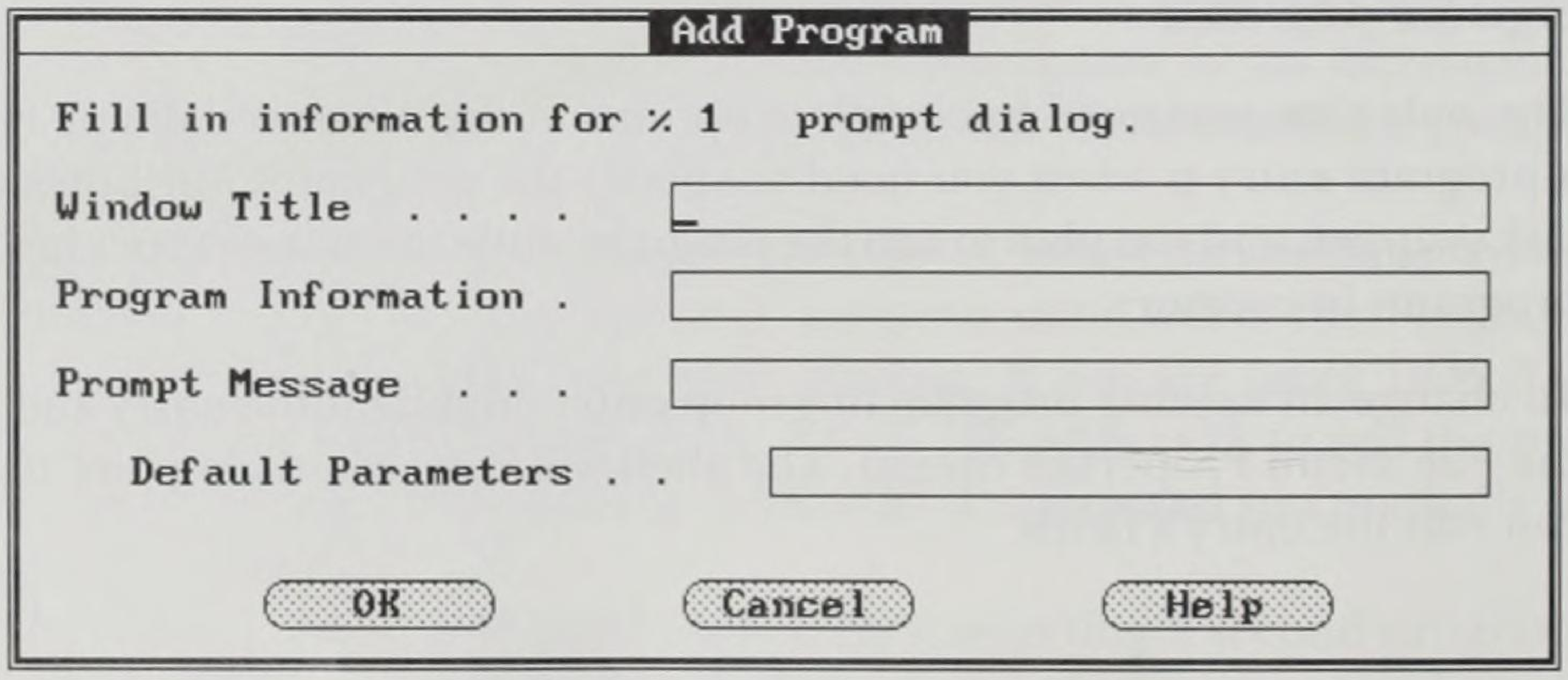
Startup Directory . .

Application Shortcut Key

[X] Pause after exit

Password . .

After you complete the entry's other fields and select OK, the shell will display the following dialog box, prompting you to define the dialog box for use with %1.



Add Program

Fill in information for % 1 prompt dialog.

Window Title . . . . _

Program Information .

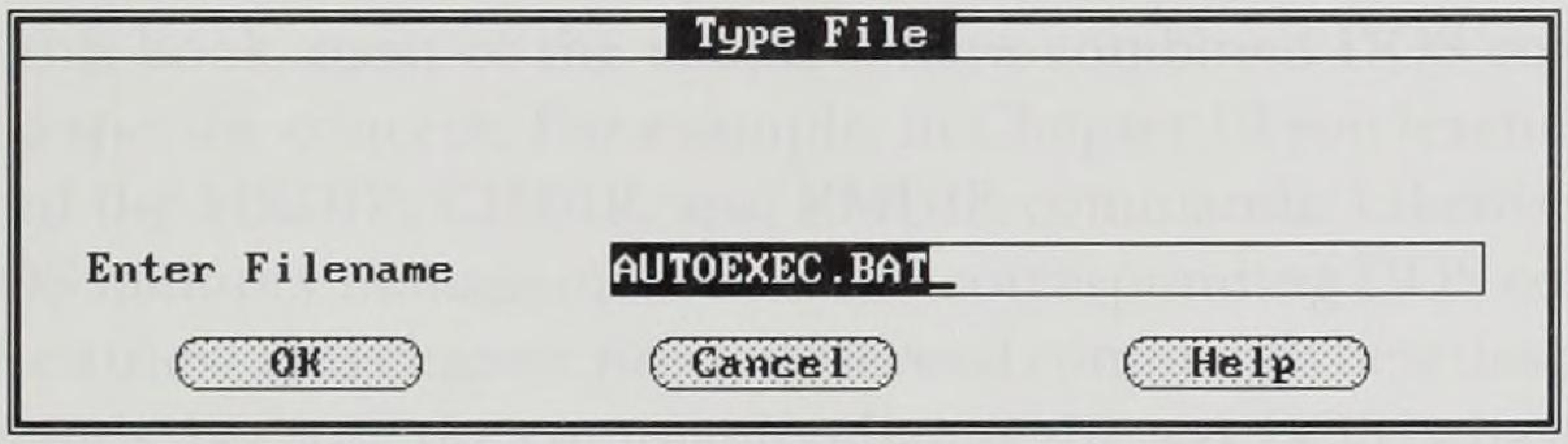
Prompt Message . . .

Default Parameters . .

The Window Title lets you specify up to 23 characters for the dialog box to display in its title bar. In this case, you might simply use the program entry name Type File. The Program Information line lets you specify up to 106 characters of instructions that appear beneath the program title. The Prompt Message specifies an 18-character prompt the shell displays to the left of the box within which the user enters the information desired. In this case, you might use "Enter Filename." The Default Parameters option lets you specify the dialog box's default value. In the case of a disk operation, you might specify a drive letter or use the shell's predefined %L parameter, which corresponds to the parameter used the last time the user selected the program. In the case of a file operation, you use the shell's predefined %F value that uses the currently selected file. If you click on the Help option, the shell lets you type in online help text specific to this dialog box.

If your program's command line contains more than one replaceable parameter, the shell will display the Add Program dialog box, previously shown, requesting you to design each box.

If the user later selects the Type File program entry with the AUTOEXEC.BAT filename selected, the shell will display the dialog box shown here:



AUTOEXEC.BAT

If a program requires several parameters, the shell will display the custom dialog box you created for each.

By creating dialog boxes and providing default values in this way, you can make running commands from within the DOS shell very straightforward.

DOSSHELL.INI

Each time you make changes to the shell, such as confirmation settings, colors, or changes to program groups, the shell stores your changes in the file DOSSHELL.INI, which resides in the DOS directory. When you start the shell, the shell uses this file to establish your settings. DOSSHELL.INI is an ASCII file whose contents you can display, print, or even edit. Before you edit this file, however, make sure you make a copy of the file's original contents. By studying this file you can learn how to change several shell settings, as well as how to create your own color definition.

Key Terms

Program group A collection of related programs, such as a word processor, spell checker, and thesaurus.

Replaceable parameter A parameter value within a shell dialog box definition with the value %1 through %9, for which the user can specify the desired value when the shell displays the dialog box, prior to running a program.

Chapter 41

DOS System Commands

Throughout this book, most of the chapters have combined DOS commands with discussion of a specific concept. For example, in Chapter 10 you learned about DOS directories and the MKDIR, CHDIR, and RMDIR commands. Likewise, Chapter 32 was about DOS memory management and the corresponding DOS commands and CONFIG.SYS entries. This chapter presents several commands best described as DOS system commands, because the commands change the way DOS behaves. When you finish this chapter, the only command left to discuss is FDISK, which will be presented in Chapter 43.

Customizing Your System Prompt

In Chapter 10 you used the command PROMPT PG to set the system prompt to the current drive and directory followed by a greater-than sign. The symbols $P and $G are *metacharacters*; characters the PROMPT command translates to produce a result. In this case, PROMPT translates $P into the current drive and directory and $G into the greater-than symbol. The PROMPT command supports the metacharacters described in Table 41-1.

Metacharacter	Translation
$B	DOS pipe symbol, |
$D	Current date
$E	ASCII escape character
$G	Greater-than symbol, >
$H	Backspace character, which erases the previous character displayed by prompt
$L	Less-than symbol, <
$N	Current disk drive letter
$P	Current drive letter and directory
$Q	Equal sign, =
$T	CURRENT time
$V	DOS version
$$	Dollar sign, $
$_	Carriage return and linefeed characters, which advance the cursor to the next line

Table 41-1. The PROMPT Metacharacter Translations

By default, DOS will display the current drive letter followed by a greater-than symbol for the prompt.

```
C>
```

Most users, however, place the PROMPT PG command in their AUTOEXEC.BAT file so that the system prompt will display the current drive and directory.

Depending on your preferences, you might want to change your prompt slightly, using the techniques discussed next. Regardless of the eventual prompt you select, I strongly recommend you include the $P metacharacter to display the current drive and directory.

The following command changes the system prompt to the current date, followed by a dollar sign:

```
C:\> PROMPT  $D$$ <ENTER>

Mon 04-05-93$
```

If you press ENTER several times, your system prompt will remain unchanged.

```
C:\> PROMPT  $D$$ <ENTER>\line
Mon 04-05-93$ <ENTER>
Mon 04-05-93$ <ENTER>
Mon 04-05-93$ <ENTER>
```

Use the SET command to display the current environment settings.

```
Mon 04-05-93$  SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS;C:\DISKUTIL;C:\WINDOWS
PROMPT=$D$$
```

Each time it completes a command, DOS searches the environment for the PROMPT entry. If the entry exists, DOS uses it to assign the system prompt. If the PROMPT entry does not exist in the environment, DOS uses the default drive letter and greater-than symbol for the prompt.

Knowing that DOS searches the environment in this manner, you actually have two ways to set the DOS prompt: using the PROMPT command or using SET PROMPT=.

If you invoke PROMPT without specifying a desired prompt, DOS selects its default PROMPT.

```
Mon 04-05-93$  PROMPT <ENTER>

C>
```

By combining the $D (date), $T (time), $_ (carriage return and line feed), $P (drive and directory) and $G (greater-than symbol), the following PROMPT command creates a prompt that spans three lines, displaying the date, time, current drive, and directory:

```
C> PROMPT  $D$_$T$_$P$G <ENTER>

Mon 04-05-93
11:03:45.55
C:\>
```

In addition to letting you use metacharacters, PROMPT lets you include text in your prompt. The following PROMPT includes the words "Current Time," followed by the system time:

```
C:\> PROMPT Current Time $T

Current Time 11:04:33.63
```

In this case, the prompt includes minutes and seconds in the system time. Using the $H backspace character you can erase the seconds and fractions of a second from the prompt, as shown here:

```
C:\> PROMPT Current Time $T$H$H$H$H$H$H <ENTER>
```

Although such prompts can be entertaining, the most useful and the most simple prompt is PG.

Enabling Extended CONTROL-BREAK Checking

As you have learned, DOS lets you end commands and batch files by pressing the CTRL-C or CTRL-BREAK keyboard combinations. By default, each time DOS writes a letter to the screen, printer, or auxiliary device (COM1), or when it reads a letter from the keyboard or auxiliary device, DOS checks to see if you have pressed CTRL-BREAK to end the command. If you have, DOS ends the command and redisplays its prompt. If you have not pressed CTRL-BREAK, DOS continues the command's execution.

Depending on the commands you are running, considerable time can pass between the times DOS performs each one of these operations. After you press CTRL-BREAK to end the command, you'll have to wait until DOS acknowledges it. If the program never performs one of these operations, you will have to wait until the program ends.

The BREAK command lets you enable and disable extended CTRL-BREAK checking. When you enable extended checking, you increase the number of times DOS will check for a CTRL-BREAK. Actually, DOS will check for a CTRL-BREAK each time it completes any operation for the command, such as file input or output, or even a date or time operation. Because DOS checks for CTRL-BREAK more often, you'll have to wait less time for DOS to acknowledge your CTRL-BREAK and end the current command. Unfortunately, because DOS spends more time checking for CTRL-BREAK, it spends less time running your program. As a result, your programs will run slower and your system performance will decrease. Because this extended checking slows your system down, most users leave it disabled.

If you invoke BREAK with no command-line parameters, BREAK will display the current state of extended CTRL-BREAK checking (either on or off) as shown here:

```
C:\> BREAK <ENTER>
BREAK is off
```

To enable extended CTRL-BREAK checking, invoke BREAK with the word ON as shown here:

```
C:\> BREAK ON <ENTER>
```

DOS will then check for a CTRL-BREAK more often. To disable CTRL-BREAK checking, invoke BREAK with the word OFF as shown here:

```
C:\> BREAK OFF <ENTER>
```

Chapter 24 discussed the CONFIG.SYS BREAK entry. By default, DOS starts with extended CTRL-BREAK checking turned off. Using the CONFIG.SYS BREAK entry, you can enable extended CTRL-BREAK checking as soon as the system starts. In general, the only users who might want to enable extended CTRL-BREAK checking are computer programmers who are writing and testing programs.

If you invoke BREAK and BREAK displays its status as already ON, check your CONFIG.SYS for a BREAK=ON entry, and your AUTOEXEC.BAT for a BREAK ON command. If either is present, remove it and restart your system. As a result, you will improve your system performance. If you want to enable extended checking before you run specific commands, do so, but then turn the checking off as soon as the command completes.

```
C:\> BREAK  ON <ENTER>

C:\> SOMECOMM <ENTER>

C:\> BREAK  OFF <ENTER>
```

Using EXIT to Resume a Program

Many programs, such as a word processor or a shell interface like the DOS shell, let you temporarily leave the program to issue commands at the DOS prompt. Using the DOS shell discussed in Chapter 39 as an example, pressing SHIFT-F9 lets you temporarily exit the shell to DOS.

Temporarily exiting a program to the DOS prompt is different than ending the program. When you end the program, you must first save your files. To use the program again, you must invoke it from the DOS prompt and reopen the files. When you temporarily exit a program to DOS, you leave things within the program just as they were—files that you have open can remain open. After you finish executing the commands you left the program to invoke, use the EXIT command to return to the program, right where you left off.

```
C:\> EXIT <ENTER>
```

The only time you will normally use the EXIT command is to return back to a program after temporarily exiting to the DOS prompt.

WARNING

Do not turn off your computer or restart DOS before you return to the program, save your files, and end the program, returning to the DOS prompt. If you don't end the program correctly, you may lose changes you have made to files.

Online Help for DOS 5 Commands

In DOS 5 you can get a brief summary of a command's function, parameters, and switches by invoking the command with the /? switch, as shown here:

```
C:\> XCOPY /? <ENTER>
Copies files (except hidden and system files) and directory trees.

XCOPY source [destination] [/A | /M] [/D:date] [/P] [/S [/E]]
  [/V] [/W]

   source       Specifies the file(s) to copy.
   destination  Specifies the location and/or name of new files.
  /A            Copies files with the archive attribute set,
                  doesn't change the attribute.
```

```
/M           Copies files with the archive attribute set,
               turns off the archive attribute.
/D:date      Copies files changed on or after the specified date.
/P           Prompts you before creating each destination file.
/S           Copies directories and subdirectories except empty ones.
/E           Copies any subdirectories, even if empty.
/V           Verifies each new file.
/W           Prompts you to press a key before copying.
```

If you can't remember which command you need, you can type **HELP** at the DOS prompt. The HELP command will display each of the DOS commands and a one-line summary, as shown here:

```
C:\> HELP <ENTER>
For more information on a specific command, type HELP command-name
APPEND   Allows programs to open data files in specified direc-
          tories as if they were in the current directory.
ASSIGN   Redirects requests for disk operations on one drive to
          a different drive.
ATTRIB   Displays or changes file attributes.
BACKUP   Backs up one or more files from one disk to another.
BREAK    Sets or clears extended CTRL-C checking.
CALL     Calls one batch program from another.
CD       Displays the name of or changes the current directory.
 :          :
 :          :
UNDELETE Recovers files which have been deleted.
UNFORMAT Restores a disk erased by the FORMAT command or restruc-
          tured by the RECOVER command.
VER      Displays the MS-DOS version.
VERIFY   Tells MS-DOS whether to verify that your files are writ-
          ten correctly to a disk.
VOL      Displays a disk volume label and serial number.
XCOPY    Copies files (except hidden and system files) and direc-
          tory trees.
```

The HELP command behaves like the MORE command. If you want to see the next screenful of commands, press any key; use CTRL-C to end the command.

In addition to listing each command, HELP lets you display help text for individual commands. For example, the following command displays help on the subject of the COPY command.

```
C:\> HELP COPY <ENTER>
Copies one or more files to another location.

COPY [/A | /B] source [/A | /B] [- source [/A | /B] [- ...]]
 [destination [/A | /B]] [/V]

  source       Specifies the file or files to be copied.
  /A           Indicates an ASCII text file.
  /B           Indicates a binary file.
  destination  Specifies the directory and/or filename for the
                  new file(s).
  /V           Verifies that new files are written correctly.

To append files, specify a single file for destination, but
multiple files for source (using wildcards or file1-file2-file3
format).
```

The information HELP displays is identical to what you would see if you invoke COPY using the /? switch. For a more complete online help for all versions of DOS, see the order form for the *DOS HELP* program that is offered at the beginning of this book.

Online Help for DOS 6 Commands

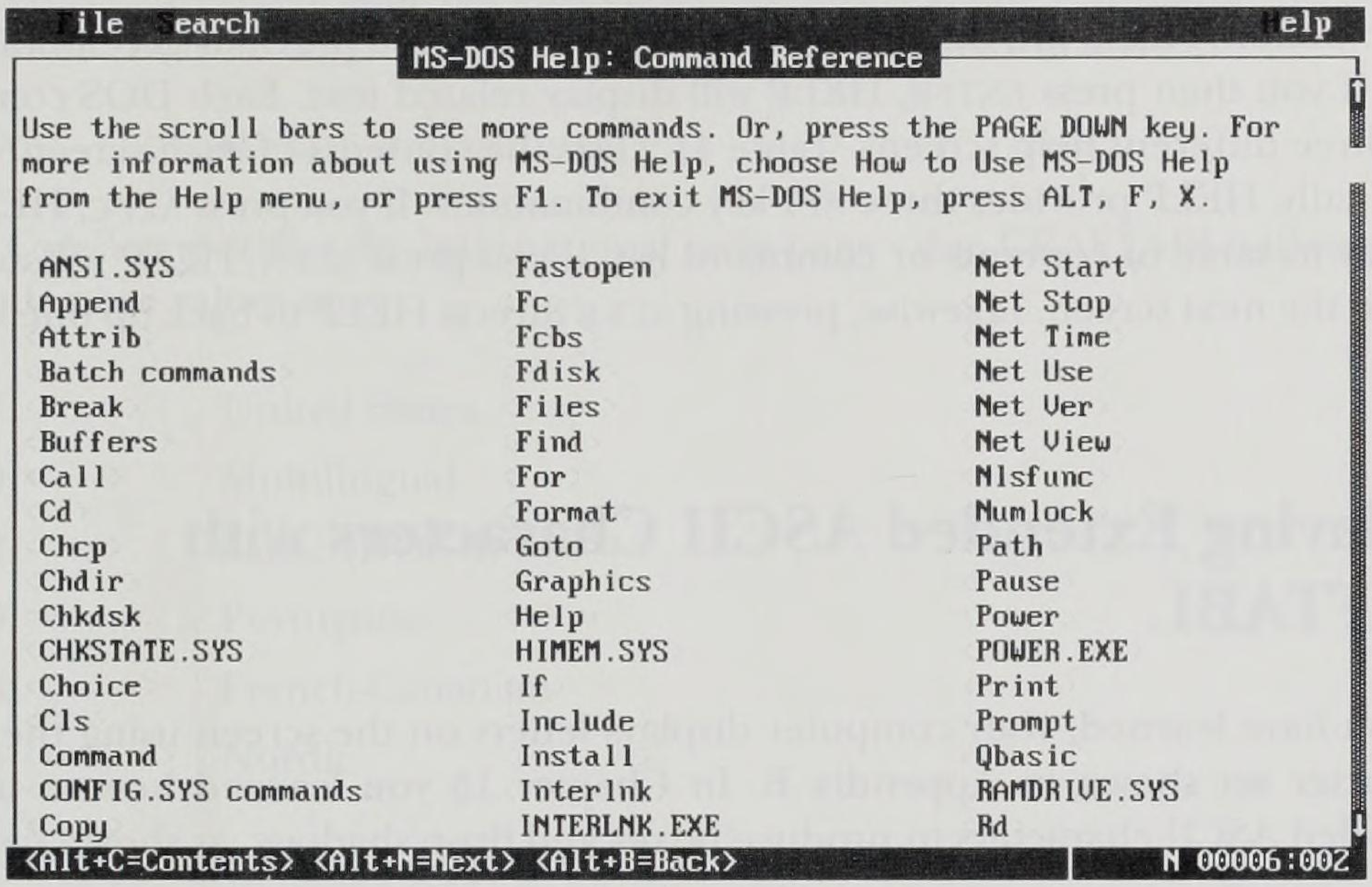
If you are using DOS 6, you can display help text for a specific command using the /? switch. For example, the following command would display information on the DOS 6 DEFRAG command:

```
C:\> DEFRAG /? <ENTER>
```

DOS 6 provides two help commands. The FASTHELP command provides the same function as the DOS 5 HELP command just discussed. The DOS 6 HELP command, invokes a menu-driven help facility, as shown in Figure 41-1.

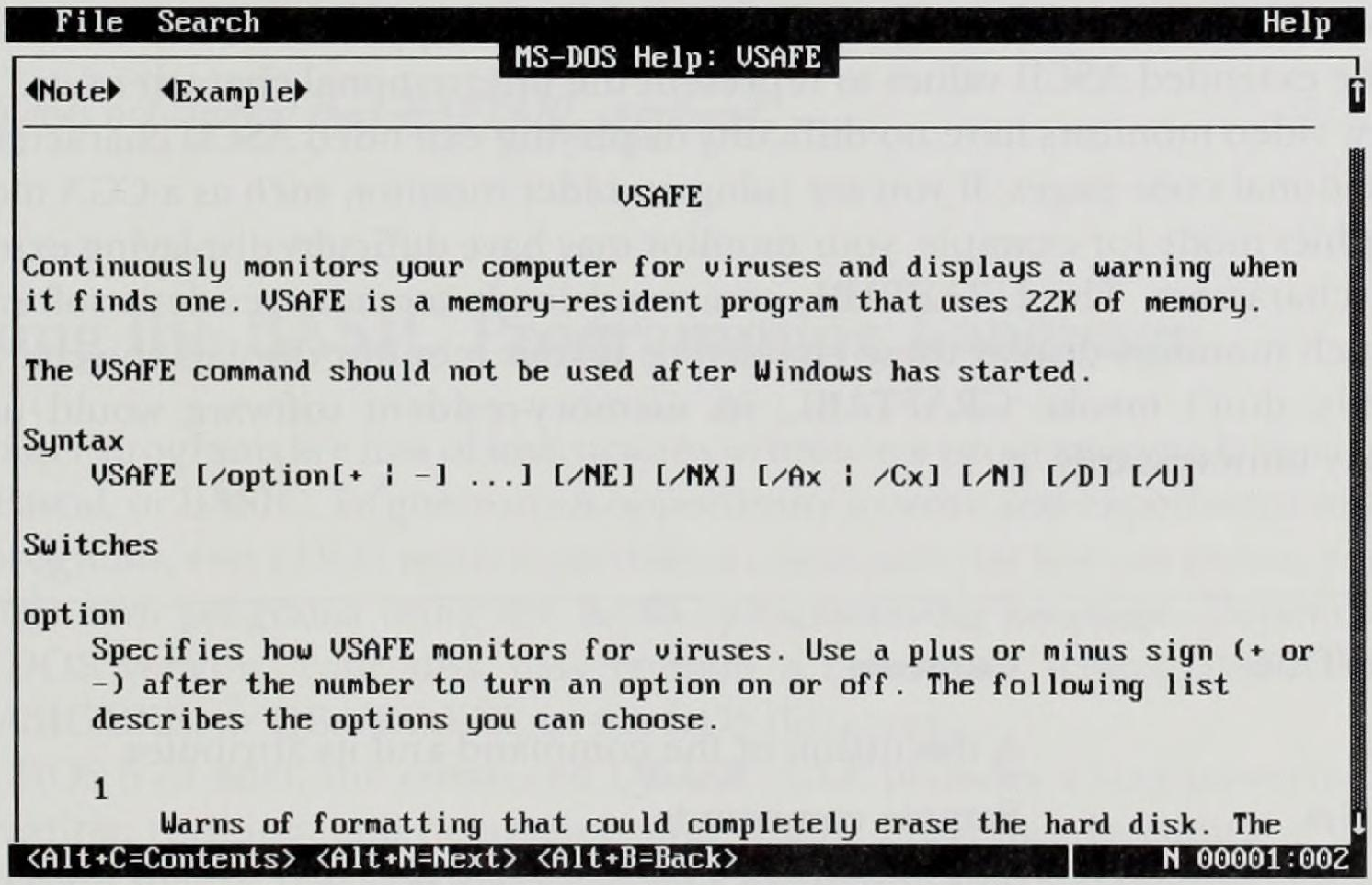
Using your keyboard arrows, TAB keys, or mouse, you can quickly select a specific command. For example, Figure 41-2 illustrates help text on the DOS 6 VSAFE command.

Using HELP's File menu, you can print the current help text or exit HELP. Using the Search command, you can quickly search for a specific topic. As you view help text, you will encounter words bracketed in green. By pressing the TAB key, you can


File Search Help
MS-DOS Help: Command Reference
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
N 00006:002

```
Use the scroll bars to see more commands. Or, press the PAGE DOWN key. For
more information about using MS-DOS Help, choose How to Use MS-DOS Help
from the Help menu, or press F1. To exit MS-DOS Help, press ALT, F, X.

 ANSI.SYS                    Fastopen                   Net Start
 Append                      Fc                         Net Stop
 Attrib                      Fcbs                       Net Time
 Batch commands              Fdisk                      Net Use
 Break                       Files                      Net Ver
 Buffers                     Find                       Net View
 Call                        For                        Nlsfunc
 Cd                          Format                     Numlock
 Chcp                        Goto                       Path
 Chdir                       Graphics                   Pause
 Chkdsk                      Help                       Power
 CHKSTATE.SYS                HIMEM.SYS                  POWER.EXE
 Choice                      If                         Print
 Cls                         Include                    Prompt
 Command                     Install                    Qbasic
 CONFIG.SYS commands         Interlnk                   RAMDRIVE.SYS
 Copy                        INTERLNK.EXE               Rd
```

Figure 41-1. *The DOS 6 HELP facility*



```
                                VSAFE

Continuously monitors your computer for viruses and displays a warning when
it finds one. VSAFE is a memory-resident program that uses 22K of memory.

The VSAFE command should not be used after Windows has started.

Syntax
    VSAFE [/option[+ | -] ...] [/NE] [/NX] [/Ax | /Cx] [/N] [/D] [/U]

Switches

option
    Specifies how VSAFE monitors for viruses. Use a plus or minus sign (+ or
    -) after the number to turn an option on or off. The following list
    describes the options you can choose.

    1
        Warns of formatting that could completely erase the hard disk. The
```

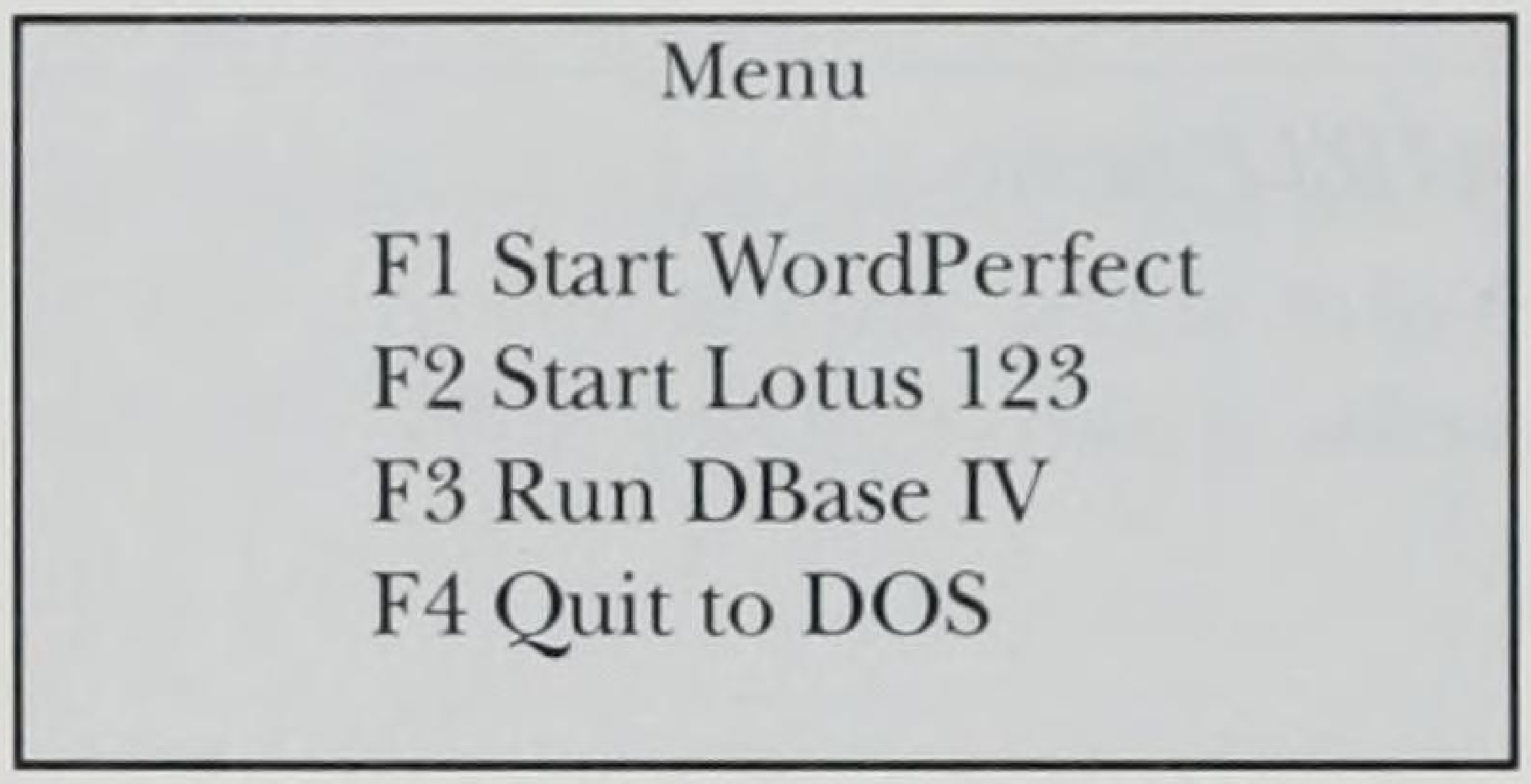
Figure 41-2. *Online help for the DOS 6 VSAFE command*

quickly advance the cursor to the text. Likewise, if you press the SHIFT-TAB keyboard combination, HELP will move the cursor backward to the previous green-bracketed text. If you then press ENTER, HELP will display related text. Each DOS command has three different help screens. Table 41-2 lists the contents of each screen.

Finally, HELP provides three ALT-key combinations. If you press ALT-C, HELP will display its table of contents or command list. If you press ALT-N, HELP will advance you to the next screen. Likewise, pressing ALT-B directs HELP to back up one screen.

Displaying Extended ASCII Characters with GRAFTABL

As you have learned, your computer displays letters on the screen using the ASCII character set shown in Appendix B. In Chapter 15 you learned how to use the extended ASCII characters to produce boxes and drop shadows, as shown here:



```
                Menu

        F1 Start WordPerfect
        F2 Start Lotus 123
        F3 Run DBase IV
        F4 Quit to DOS
```

Chapter 38 examined the code-page settings international users can use to represent their country's character set. If you examine Appendix C, you will find DOS uses the extended ASCII values to represent the international character sets.

New video monitors have no difficulty displaying extended ASCII characters and international code pages. If you are using an older monitor, such as a CGA monitor in graphics mode for example, your monitor may have difficulty displaying extended ASCII characters. The GRAFTABL command loads memory-resident software to help such monitors display these characters. If your monitor displays the characters correctly, don't invoke GRAFTABL; its memory-resident software would use up memory unnecessarily.

Screen Title	Contents
Notes	A discussion of the command and its attributes
Examples	Sample commands
Syntax	The command's format and switches

Table 41-2. *The Contents of DOS 6 HELP Screens*

Before DOS version 3.3, the format of the GRAFTABL command was simply GRAFTABL. In DOS 3.3 through DOS 5, the command's format is

```
GRAFTABL  [CodePage] [/STATUS]
```

where *CodePage* specifies the international code-page value GRAFTABL will support. The code-page values are

437	United States
850	Multilingual
852	Slavic (DOS 5)
860	Portuguese
863	French-Canadian
865	Nordic

The /STATUS switch directs GRAFTABL to display the current *CodePage* selection.

The following command, for example, provides support for the extended ASCII character set (or U.S. code page):

```
C:\> GRAFTABL  437 <ENTER>
```

Remember, the only time you need to use GRAFTABL is when your monitor does not properly display the character set while operating in graphics mode.

NOTE

DOS 6 does not support the GRAFTABL command.

Invoking the BASIC Programming Language

Computer programs are lists of instructions written in a programming language such as C, Pascal, or BASIC. To give you an opportunity to write and experiment with your own programs, every DOS version provides a command that lets you create, edit, and run your own programs using the BASIC programming language. Depending on your DOS version, your disk may contain a command named BASICA.COM, GWBASIC.EXE, or QBASIC.EXE in the DOS directory.

In DOS 5 or later, the command QBASIC.EXE provides a very powerful editor for creating, running, and debugging your BASIC programs. In addition, you may find several files in the DOS directory that contain the BAS extension. These files contain BASIC programs you can view and run.

For more information on GW-BASIC or QBASIC, refer to Osborne/McGraw-Hill's books, *GW-BASIC Made Easy* or *QBASIC Made Easy* (both by Bob Albrecht and Don Inman).

Revisiting MODE

In Chapter 13 you may have used the MODE command to redirect printer output from the default PRN device to a serial device, such as COM1. In Chapter 38 you may have used the MODE command to prepare and select an international code page. In this chapter, you will examine the rest of MODE's capabilities using MODE for printer, video, and keyboard customization.

Customizing Your Printer Output

If you are using an Epson-compatible or IBM-compatible printer, you can use the MODE command to specify the character size and number of characters per line the printer will print. For DOS versions 3.3 and earlier, the format of the MODE command is as follows:

```
MODE  LPTn[:]  [CharactersPerLine] [,[LinesPerInch] [,Retry]]
```

In DOS 4 or later, the format of the command becomes the following:

```
MODE  LPTn[:]  [cols=CharactersPerLine] [lines=LinesPerInch]
    [retry=Retry]
```

CharactersPerLine specifies either 80 or 132 characters per line. By default, most printers use 80 characters per line and either wrap or ignore characters that are past column 80. If you specify 80 characters per line, MODE will discard (ignore) any characters appearing past column 80. *LinesPerInch* specifies the number of lines, either 6 or 8, the printer will print in a vertical inch on your page. The default setting is 6. If you select 8, your printed output becomes more condensed.

Retry specifies how DOS should react to timeout errors when it tries to send information to a parallel printer and the printer is not ready. By default, MODE will wait for a period of time for the printer and then time out. Table 41-3 lists the valid retry options. These options are more meaningful to DOS than to a user.

When you specify a retry option, MODE will load memory-resident software that monitors the printer and its status.

NOTE

The retry option P directs MODE to retry sending data to the printer until it accepts the data. If your printer is slow, it will eventually be ready for more data. If your printer experiences an error, MODE will remain in its continuous loop, trying to send data to the printer (apparently hanging your system) until you break the retry loop by pressing the key combination, CTRL-C.

Retry Option	Function
E	Returns an error status
B	Returns a busy status
P	Retries the operation until the printer accepts data
R	Returns a ready status
N or None	Takes no action

Table 41-3. *The Retry Options for Printer Timeout Errors*

The following command, for example, selects compressed print of 132 characters per column and 8 lines per vertical inch:

```
C:\> MODE  LPT1:  132,8  <ENTER>
```

To direct DOS to continuously retry printer operations until the operation is successful, include the *p* option, as shown here:

```
C:\> MODE  LPT1:  132,8,p  <ENTER>
```

If you are using DOS 4 or later, you can issue the same command as follows:

```
C:\> MODE  LPT1:  COLS=132  LINES=8  RETRY=P <ENTER>
```

Not all printers let you set their printer attributes using MODE in this way. If your printer does not, your printer should support different escape sequences to achieve the same result. Refer to your printer documentation for more information.

Most newer programs provide built-in software to manage the printer, specify the fonts, etc. Using these programs, you probably won't need to use MODE to customize your printer. However, if a program has difficulty printing, you might experiment with MODE. In Chapter 34 you used escape sequences with the ANSI device driver to enhance your screen and keyboard capabilities. Most printers have a set of predefined escape sequences you can print to select various fonts, page layouts, and other printer attributes. For more information on printer escape sequences, refer to the documentation that accompanied your printer.

Using MODE to Set Your Keyboard's Response Rate

If you are using DOS 4 or later, MODE lets you set your keyboard's *typematic rate*, the rate at which DOS repeats a character if you leave the key depressed. The higher the value you assign to the typematic rate, the faster DOS repeats characters.

Not all keyboards support the typematic value.

NOTE

The format of this MODE command is as follows:

```
MODE  CON[:]  [RATE= RepeatRate DELAY=DelayTime]
```

The CON device corresponds to your keyboard.

A *RepeatRate* value of 1 lets the keyboard repeat 2 characters per second, and the value 32 specifies 30 characters per second. By default, DOS uses a *RepeatRate* value of 20 (10 characters per second) for PC-AT keyboards and 21 (11 characters per second) for PS/2 keyboards. Table 41-4 lists the approximate characters per second for different *RepeatRates*.

Repeat Rate	Characters per Second	Repeat Rate	Characters per Second
1	2.0	17	8.0
2	2.1	18	8.6
3	2.3	19	9.2
4	2.5	20	10.0
5	2.7	21	10.9
6	3.0	22	12.0
7	3.3	23	13.3
8	3.7	24	15.0
9	4.0	25	16.0
10	4.3	26	17.1
11	4.6	27	18.5
12	5.0	28	20.0
13	5.5	29	21.8
14	6.0	30	24.0
15	6.7	31	26.7
16	7.5	32	30.0

Table 41-4. *Keyboard Typematic Repeat Rate Values and their Corresponding Numbers of Characters per Second*

DelayTime is a value from 1 to 4 that specifies the amount of time DOS will wait after you first press the key before it begins to repeat characters. Table 41-5 summarizes the delay settings.

For the fastest keyboard response, use the following command:

```
C:\> MODE  CON:  RATE=32  DELAY=1 <ENTER>
```

If your keyboard begins repeating keys you don't intend to repeat, try adjusting the delay value first to give your fingers a little more time to release the key.

Customizing Your Video Display

Depending on your video display type, the MODE command lets you specify the number of characters per row and number of rows per screen. Before DOS 4, the format to select a screen setting uses the MODE command followed by one of the display modes listed in Table 41-6.

The following MODE command, for example, selects 40 columns per row:

```
C:\> MODE  40 <ENTER>
```

To restore the screen to 80 columns, issue the following MODE command.

```
C:\>  MODE  80 <ENTER>
```

Beginning with DOS 4, MODE lets you specify the number of columns and rows as follows:

```
MODE  CON[:] [COLS=Columns] [Lines=Rows]
```

Delay Time Value	Delay in Seconds
1	1/4 second
2 (default)	1/2 second
3	3/4 second
4	1 second

Table 41-5. *Keyboard Typematic Delay Time Values and their Corresponding Delays in Seconds*

Display Mode	Display Setting
40	40 characters per row
80	80 characters per row
BW40	Color disabled on a color adapter, 40 characters per row
BW80	Color disabled on a color adapter, 80 characters per row
CO40	Color enabled, 40 characters per row
CO80	Color enabled, 80 characters per row
MONO	Monochrome adapter, 80 characters per row

Table 41-6. *Video Display Settings Supported by MODE*

The *Columns* value must be either 40 or 80. For CGA systems, the *Rows* value must be 25. For EGA, rows can be 25 or 43. Finally, for VGA systems, rows can be 25, 43, or 50. To set the number of rows using MODE, you must have previously installed the ANSI.SYS device driver, as discussed in Chapter 34.

The following command sets an EGA or VGA monitor to 43 rows:

```
C:\> MODE CON LINES=43 <ENTER>
```

Similarly, this command sets a VGA monitor to 50 rows:

```
C:\> MODE CON LINES=50 <ENTER>
```

If you are using an older monitor, you may need to shift characters to the left or right to view the characters on the screen. MODE lets you install memory-resident software that will automatically shift the characters. To shift characters to the left or right, use the following MODE command format:

```
MODE DisplayMode,L|R[,T]
```

DisplayMode specifies one of the video adapter types listed in Table 41-6. The entry L|R specifies that you must select either the letter L to shift characters to left or R to shift characters to the right. The optional T value directs MODE to display a test pattern of the values 0 through 9. The following example illustrates a right-shift operation.

```
0123456789012345678901234567890123456789012345678901234567890123456789
Do you see the leftmost 0 (Y/N)?
```

In this case, if you type N to MODE's prompt, MODE will shift the letters to the right. When you eventually respond with Y, the MODE command will end MODE, loading memory-resident software that keeps the characters shifted until you turn off your computer or restart DOS.

Configuring Your Serial Port

You learned in Chapter 13 that your computer has one or more serial or parallel ports, as shown in Figure 41-3.

The parallel ports (LPT1, LPT2, and LPT3) normally connect a printer to your computer. The serial ports (COM1 through COM4) can connect printers, a mouse, or a modem, which lets your computer communicate with another computer over telephone lines.

In general, parallel devices send data over eight lines. A serial device, on the other hand, sends data over one line. Each ASCII character requires eight bits (or binary digits) to uniquely distinguish the character from another. Using the word "DOS," for example, the binary digits required to represent the three letters become

D	O	S
01000100	01001111	01010011

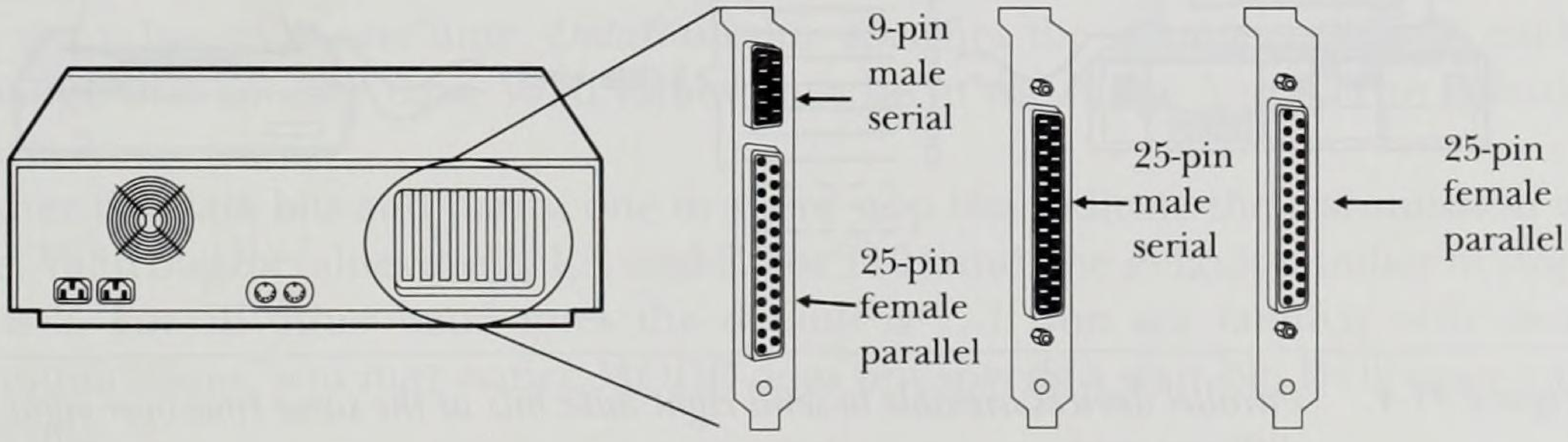


Figure 41-3. *Your computer has one or more serial or parallel ports*

Because a parallel device can send data over eight lines at one time, each line sending a binary digit, your computer can send the word "DOS" to your printer as shown in Figure 41-4.

A serial device, on the other hand, can only send information over one wire, one bit at a time. Figure 41-5 illustrates how your computer would send the word "DOS" to a serial printer.

Because parallel printers can send more data at one time (in parallel), they are much faster than serial devices. If you look at a parallel or serial printer cable, however, you'll find that both cables contain many pins. In addition to the eight wires for data, parallel devices use wires to control the data transmission. For example, if your printer's memory buffer becomes full, it can use one of these wires to signal your computer to quit sending data until it is ready.

Although serial devices also have control wires available to them, most serial devices don't use them. Instead, your computer and the serial device agree how fast they will communicate, and how many bits of information they will exchange at one time. These values specify a *communication protocol.*

When you attach a printer to a serial port, the printer must know how fast and how much data the computer will send it. In most cases, the device determines the protocol. Using the MODE command, you tell the serial port the protocol values to use. Before DOS 4, the format to set a serial port's communication parameters is

```
MODE COMn[:] BaudRate[,Parity[,DataBits[,StopBits[,P]]]]
```

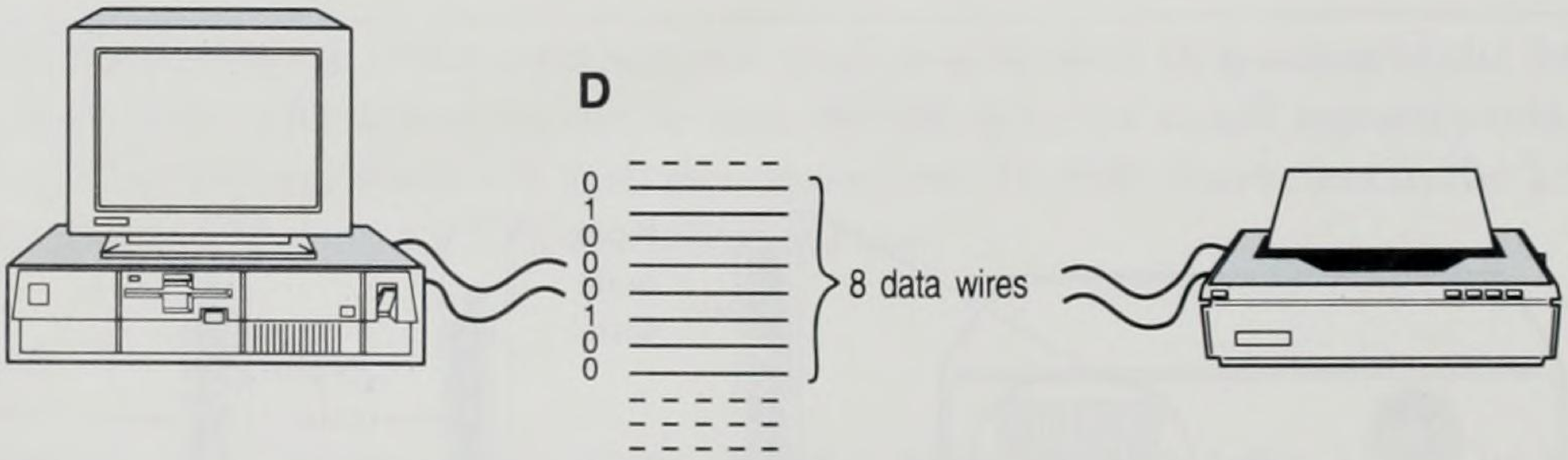


Figure 41-4. *Parallel devices are able to send eight data bits at the same time over eight different wires*

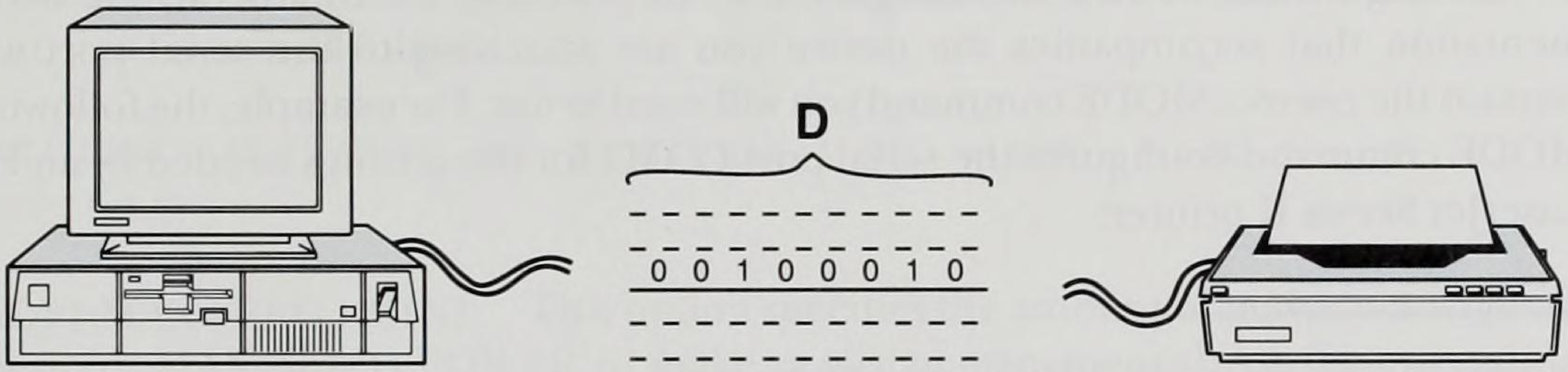


Figure 41-5. *Serial devices send one bit of data at a time over one wire*

Beginning with DOS 4, the MODE command becomes

```
MODE COMn[:] [baud=BaudRate] [parity=Parity] [data=DataBits]
   [StopBits] [retry=Retry]
```

BaudRate specifies the speed at which the computer and the device will communicate in bits per second. Valid baud rates are 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200 bits per second. MODE only requires you to specify the first two baud-rate digits.

To help the computer or device recognize single bit errors, an extra parity bit is often sent immediately following the data bits. When the computer or device gets the data, it counts the number of binary digits that had the value 1. The result will either be an even or odd number. If the computer and device agree to use even parity, and the number of 1 digits is odd, a transmission error occurs. The same is true for odd parity and an even number of 1 digits. You must specify the letter E for even parity, O for odd parity, an N for no parity, S for space, and M for mark. Space parity always assigns the parity bit the value 0, while mark parity assigns the value 1. The default setting is even parity. The computer and device must agree on the amount of data to exchange at one time. *DataBits* value specifies the number of bits in each exchange that contain data. Valid values must be in the range 5 to 8. The default value is 7.

After the data bits and parity, one or more stop bits indicate the transmission is done. Valid *StopBit* values are 1, 1.5, and 2. For 110 baud, the default number of stop bits is 2. For all other baud rates the default is 1. If you are familiar with data communications, you may notice MODE does not specify a start bit. DOS does not need one.

Finally, the retry option specifies how MODE will handle device busy errors. Valid retry options are listed in Table 41-3.

Although using MODE to configure a serial port may seem difficult, the documentation that accompanies the device you are attaching to the serial port will contain the correct MODE command you will need to use. For example, the following MODE command configures the serial port COM1 for the settings needed by an HP LaserJet Series II printer:

```
C:\> MODE COM1 9600,N,8,1,p <ENTER>
```

If you need to configure a serial port with MODE, place the command provided in the device documentation in your AUTOEXEC.BAT file and don't worry about the command after that.

Displaying a Device's Current Status

As you have learned, the MODE command lets you configure your printer, screen display, keyboard, and serial devices. Beginning with DOS 4, you can use MODE /STATUS to display the current device settings. For example, the following MODE command displays the status of the serial port COM1:

```
C:\> MODE  COM1 /STATUS <ENTER>

Status for device COM1:
-------------------
Retry=B
```

If you don't specify a device name, as shown in the following, MODE will display the status of every known device:

```
C:\> MODE /STATUS <ENTER>
```

DOS 6 Advanced Power Management

One of the biggest concerns laptop users face is battery life. To increase battery life, Intel and Microsoft have recently created the advanced power management (APM) specification, which combines hardware and software to reduce the PC's power consumption during idle times. Because the APM specification is new, very few computers support it yet. If you have a newer computer that does, you can use the DOS 6 POWER command to control its power consumption. To begin, you must first install the POWER.EXE within your CONFIG.SYS file. The format of the DEVICE entry to do so is as follows:

```
DEVICE=C:\DOS\POWER.EXE [ADV[:MAX | REG | MIN] | STD |
OFF][/LOW]
```

ADV[:MAX | REG | MIN] This option specifies the amount of power management desired. MAX directs POWER to make power management the primary consideration, possibly at the cost of application performance. REG is the default setting and directs POWER to balance application performance with power management. MIN directs POWER to only reduce power consumption when applications and devices are idle.

STD This option directs POWER to only use those APM features your hardware supports. If your hardware does not support APM, power management is turned off.

OFF This option directs POWER to turn off power management.

/LOW This option directs POWER to load its memory-resident software in lower (conventional) memory. By default, POWER loads itself into upper memory.

The following CONFIG.SYS entry loads maximum support for battery power conservation:

```
DEVICE=C:\DOS\POWER.EXE ADV:MAX
```

After you have installed the POWER.EXE device driver, you can use the POWER command to control the power management settings. The format of the POWER command is as follows:

```
POWER [ADV: [MAX | REG | MIN] | STD | OFF]
```

The following command selects minimum power conservation:

```
C:\> POWER ADV:MIN <ENTER>
```

To display the current power management setting, invoke POWER with no arguments, as shown here:

```
C:\> POWER <ENTER>
```

SUMMARY

PROMPT

The PROMPT command lets you customize your system prompt by combining text with one or more predefined metacharacters. The PROMPT command stores the current prompt in the PROMPT environment entry. Each time DOS completes a command, DOS uses the environment entry to assign the current prompt. If the PROMPT entry does not exist, DOS uses its default prompt of the current drive letter and greater-than symbol. The following PROMPT command directs DOS to display the current drive and directory:

```
C> PROMPT $P$G <ENTER>
```

BREAK

The BREAK command lets you enable and disable extended CTRL-BREAK checking. By default, DOS checks for CTRL-BREAK each time it writes to the screen, printer, or auxiliary device, or when it reads from the keyboard or auxiliary device. When you enable extended CTRL-BREAK checking, DOS will check for a CTRL-BREAK each time it performs any function. Because extended CTRL-BREAK checking increases the number of times DOS checks for a CTRL-BREAK, DOS will recognize your CTRL-BREAK much sooner, ending the command. However, by enabling this extended checking, the amount of extra time DOS spends checking for a CTRL-BREAK decreases the amount of time DOS lets your program run. As a result, your programs run slower and your system performance suffers. To maximize your system performance, you should leave extended CTRL-BREAK checking off using the following command:

```
C:\> BREAK OFF <ENTER>
```

EXIT

The EXIT command lets you return to a program such as the DOS shell after temporarily exiting the program to issue commands at the DOS prompt. So that you don't have to save your files and end the program, only to later restart the program and reopen the files, many programs let you exit the program temporarily so you can issue commands at the DOS prompt.

SUMMARY

When you are ready to resume the program's operation, invoke the EXIT command as shown here:

```
C:\> EXIT <ENTER>
```

HELP

In DOS 5 and later, the HELP command lets you display a list of the DOS commands and a one-line description of each. To view the Help command's primary list, invoke HELP as shown here:

```
C:\> HELP <ENTER>
```

In addition, if you include the command's name in the HELP command line, HELP lets you display information about specific commands. Using this technique, the following command displays help on UNDELETE:

```
C:\> HELP  UNDELETE <ENTER>
```

GRAFTABL

Older video monitors are unable to correctly display the extended ASCII character set in certain graphics modes. The international code pages, for example, all use extended ASCII values. The GRAFTABL command loads memory-resident software to help the monitor correctly display these characters. If you are an international DOS user using the multilingual code page (850), you would execute the following command:

```
C:\> GRAFTABL  850 <ENTER>
```

DOS 6 Power Management

Starting with version 6, DOS provides the POWER command, which enables users of laptops that conform to the APM specification to optimize the management of the battery's power.

Hands On

Using the BREAK command, determine if you currently have extended CTRL-BREAK checking enabled (on) or disabled (off).

```
C:\> BREAK <ENTER>
```

If extended CTRL-BREAK checking is on, you can improve your system performance by turning it off. Make sure your CONFIG.SYS file does not have a BREAK=ON entry and that your AUTOEXEC.BAT file does not turn extended checking on using BREAK ON.

Next, using the following PROMPT command, change your system prompt to display the current DOS version number followed by two blank lines, and then the current drive and directory followed by a greater-than sign:

```
C:\> PROMPT  $V$_$_$_$P$G <ENTER>
```

Use the SET command to display the current environment entries.

```
MS-DOS Version 6.0


C:\> SET <ENTER>
```

Note the PROMPT entry. Using SET, remove the entry.

```
MS-DOS Version 6.0


C:\> SET  PROMPT= <ENTER>
```

Note that DOS displays its default prompt of the current drive followed by a greater-than symbol. Use the following PROMPT command to display the current drive and directory followed by a greater-than sign:

```
C>  PROMPT $P$G <ENTER>
```

If you are using DOS 5, issue the following HELP command to display a summary of the available commands:

```
C:\> HELP <ENTER>
```

When HELP completes, invoke HELP with MODE to display information about the command.

```
C:\> HELP  MODE <ENTER>
```

If you are using DOS 4 or later, hold down the D key and note how fast DOS repeats the character. Next, issue the following MODE command to maximize your keyboard typematic rate.

```
C:\> MODE  CON  RATE=32  DELAY=1 <ENTER>
```

Hold down the D key once again and note the keyboard's increase in responsiveness. If you are a fast typist, you might consider placing this command within your AUTOEXEC.BAT file.

Review

1. What is extended CTRL-BREAK checking?
2. How does the CONFIG.SYS BREAK entry differ from the BREAK command?
3. What is a metacharacter?
4. What is a typematic rate?
5. What is a serial communication protocol?
6. What is APM?

Answers

1. DOS lets you end most programs and batch files by pressing CTRL-BREAK. By default, each time DOS reads from the keyboard or auxiliary device or writes

to the screen, printer, or auxiliary device, DOS checks to see if you have pressed CTRL-BREAK to end the program. If you have a program that does not perform one of these actions on a regular basis, it may take DOS considerable time before it acknowledges your CTRL-BREAK. Extended CTRL-BREAK checking extends the number of times DOS will check for a CTRL-BREAK. Using the BREAK command, you can enable and disable extended CTRL-BREAK checking. When extended checking is enabled, DOS will check for a CTRL-BREAK each time it completes an operation, which means DOS will recognize and respond to your CTRL-BREAK much sooner. Because DOS spends more time testing for CTRL-BREAK, your programs run slower, which decreases your system performance. Most users should leave extended CTRL-BREAK checking off.

2. The CONFIG.SYS BREAK entry and the BREAK command both let you enable and disable CTRL-BREAK checking. The CONFIG.SYS BREAK entry lets you specify the state of extended checking DOS will use from the moment your system starts. You can later override this initial setting using the BREAK command.

3. A metacharacter is a predefined character that follows a dollar sign character whose meaning the PROMPT command translates. For example, PROMPT translates the $P metacharacter into the current drive and directory.

4. The typematic rate controls how fast your keyboard will repeat a key when you leave the key pressed. Beginning with DOS 4, the MODE command lets you set your keyboard's typematic rate. The faster the rate, the faster DOS repeats keys, improving your keyboard's responsiveness.

5. Your computer and serial device must agree on the speed and data format they will use to exchange information. The communication protocol specifies such characteristics as the baud rate (speed), as well as the number of data bits, stop bits, and parity.

6. APM is an abbreviation for advanced power management. APM is a specification that defines how the PC uses power during idle time. Several newer laptop computers provide chips that support APM. The DOS 6 POWER command enables you to control the APM power chips, and hence your PC's power consumption.

Advanced Concepts

This section completes the discussion of all the DOS commands with the exception of FDISK (Chapter 43). For more specifics on each DOS command, turn to the Command Reference at the end of Chapter 45.

Changing the Terminal Device

The CTTY command has been with DOS since version 1. It remains one of the least used commands. CTTY directs DOS to change its source of commands and output destination from your keyboard and screen to a terminal device attached to a serial port. The format of the CTTY command is as follows:

```
CTTY COMn[:] | CON | AUX | NUL | PRN | PRN | LPTn[:]
```

Assume, for example, you have a terminal or a custom hardware device attached to your computer's serial port COM1. Using the following CTTY command, you can direct DOS to use the terminal instead of your computer's screen and keyboard.

```
C:\> CTTY  COM1: <ENTER>
```

Do not use the CTTY command unless you have a way to execute commands from the port specified. Otherwise, you will have to restart your system to continue.

If you use CTTY to access a terminal connected to a serial port, you can issue the following CTTY command from the terminal to restore normal keyboard and screen operations.

```
C:\> CTTY  CON <ENTER>
```

Transferring System Files to a Target Disk

A bootable DOS disk contains the hidden system files IO.SYS and MSDOS.SYS, as well as the DOS command processor COMMAND.COM. The easiest way to create a bootable disk is using FORMAT /S. If a disk already contains files, however, you won't want to format the disk to make it bootable because doing so would destroy the information the disk contains. The SYS command lets you transfer the files DOS needs to boot to a target disk. For example, you can use SYS to update a disk containing the startup files for an older version of DOS to a newer version.

Depending on your DOS version, how SYS behaves will differ. Before you see how SYS differs, you need to understand the processing SYS performs. First, provided the target disk has sufficient disk space, SYS copies the files IO.SYS and MSDOS.SYS to the target disk. Your disk's boot record contains a small program that loads DOS into memory. Because this startup program must be small enough to fit in the boot record, it cannot be very complex, and thus requires the file IO.SYS to be the very first entry in your disk's root directory, immediately followed by the file MSDOS.SYS. If these first two directory entries are in use, the SYS command will fail under all DOS versions prior to DOS 4, displaying the error message shown in the next example.

```
No room for system on destination disk
```

In addition, prior to DOS 4, the two files IO.SYS and MSDOS.SYS had to reside in two consecutive disk locations at the start of your disk's data area, as shown in Figure 41-6.

If you examine the FORMAT command in the Command Reference, you will find that the /B switch directs FORMAT to reserve the directory entries and disk space for these two files on a disk, but not to place the files on the disk. In the early 1980s, some computer games that ran from floppy disks would direct you to restart your system for the programs to run. Due to licensing restrictions, the software companies could not ship DOS on their floppy disks. Instead, the companies would format the floppy disks using the /B switch to leave space for DOS. The companies could then copy their files to the disk. Before you could run the program, you would have to use SYS to copy the DOS system files to the disk to create a bootable disk.

Beginning with DOS 3.3, the SYS command became a little more flexible. Version 3.3 eliminated the restriction that IO.SYS and MSDOS.SYS reside in consecutive locations at the start of the disk's data area. In DOS 4, if files already resided in the target disk's first two root directory entries, SYS moved the directory entries (provided the root directory had two unused entries) to make room for the IO.SYS and MSDOS.SYS entries.

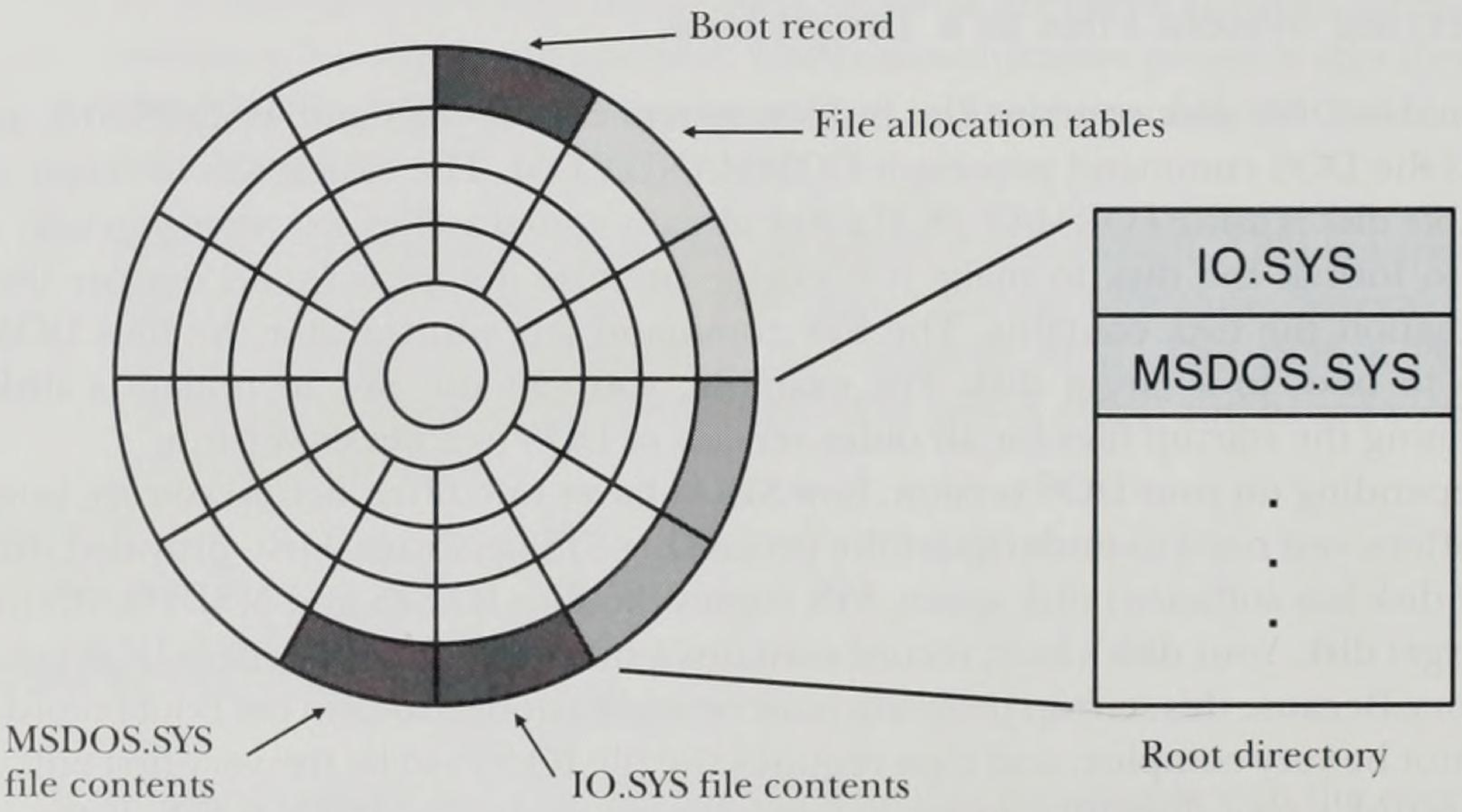


Figure 41-6. *Older DOS versions not only required IO.SYS and MSDOS.SYS to be the first two root directory entries, the files had to occupy the first locations in the disk's data area, too*

To transfer system files prior to DOS 4, the format of the SYS command is

```
SYS TargetDrive:
```

Using this format, the current drive must contain the bootable system files SYS is to transfer. Beginning with DOS 4, SYS lets you not only specify the target drive to which you want to transfer files, but also a source drive and directory containing the files to transfer.

```
SYS SourcePath: TargetDrive:
```

If you don't specify a source drive and directory, SYS uses the root directory of the current disk by default.

The following SYS command transfers a copy of the DOS system files from drive C to the floppy disk in drive A:

```
C:\> SYS A: <ENTER>
```

In DOS 4 or later, the following SYS command copies the DOS system files from the floppy disk in drive A to the floppy disk in drive B:

```
C:\> SYS A: B: <ENTER>
```

In addition to IO.SYS and MSDOS.SYS, your hard disk must have a copy of COMMAND.COM before it will boot. The DOS 5 or later SYS command automatically copies the COMMAND.COM file to the target disk for you. If you are using a DOS version earlier than 5, you will need to copy the COMMAND.COM file to the target disk.

Directing DOS 5 or later to Report a Specific Version Number

When many programs first start, they may ask DOS to tell them its version number. If, for example, a program needs a capability provided by DOS 3.3 and you are running DOS 3.2, the programs can display an error message and end. Unfortunately, if such a program has not yet been updated to know about DOS 5 or later, the program might display the same error message and end just because the program is testing for version 3.3 or 4. To let you run such programs, the SETVER command lets you specify the version number you want DOS to report to the program when the program asks. In other words, you can trick the program into thinking it's running under a different DOS version.

Before you can use SETVER to provide version number reporting, you must load the file SETVER.EXE as a device driver using the CONFIG.SYS DEVICE or DEVICEHIGH entry.

```
DEVICE=C:\DOS\SETVER.EXE
```

Assume, for example, you have a program named V33ONLY.EXE that checks for DOS version 3.3 and ends if it encounters any other version. Using the following SETVER command, you can tell DOS to report the version number 3.3 when the program V33ONLY asks

```
C:\> SETVER  V33ONLY.EXE  3.3 <ENTER>
```

SETVER will place information about V33ONLY.EXE in a table DOS reads each time it starts. SETVER will tell you it has added the new entry to the table, but that you have to restart your system for the table entry to take effect.

```
C:\> SETVER  V33ONLY.EXE  3.3  <ENTER>

WARNING - The application you are adding to the MS-DOS version
table may not have been verified by Microsoft on this version of
MS-DOS. Please contact your software vendor for information on
whether this application will operate properly under this ver-
sion of MS-DOS. If you execute this application by instructing
MS-DOS to report a different MS-DOS version number, you may lose
or corrupt data, or cause system instabilities. In that cir-
cumstance, Microsoft is not responsible for any loss or damage.

Version table successfully updated
The version change will take effect the next time you restart
your system
```

DOS provides several predefined entries for you. If you invoke SETVER without a command line, SETVER will display the current entries.

```
C:\> SETVER  <ENTER>
WIN200.BIN      3.40
WIN100.BIN      3.40
```

```
WINWORD.EXE      4.10
EXCEL.EXE        4.10
HITACHI.SYS      4.00
MSCDEX.EXE       4.00
REDIR4.EXE       4.00
NET.EXE          4.00
NET.COM          3.30
NETWKSTA.EXE     4.00
DXMA0MOD.SYS     3.30
       .
       .
       .
BAN.EXE          4.00
BAN.COM          4.00
MSREDIR.EXE      4.00
METRO.EXE        3.31
IBMCACHE.SYS     3.40
REDIR40.EXE      4.00
DD.EXE           4.01
DD.BIN           4.01
LL3.EXE          4.01
REDIR.EXE        4.00
SYQ55.SYS        4.00
SSTDRIVE.SYS     4.00
ZDRV.SYS         4.01
ZFMT.SYS         4.01
TOPSRDR.EXE      4.00
```

When you later receive the DOS 5 upgrade for one of the programs, you can remove the entry from the table using the SETVER/DELETE switch, shown here:

```
C:\> SETVER  V33ONLY.EXE  /DELETE  <ENTER>

Version table successfully updated
The version change will take effect the next time you restart
your system
```

As you can see, SETVER again displays information telling you to restart your system for the change to take effect. To suppress the display of the system restart message, you can include the /QUIET switch when you delete entries from the table.

```
C:\> SETVER  V33ONLY.EXE  /DELETE  /QUIET <ENTER>
```

Using SETVER, you can upgrade to DOS 5 or later and still use programs that only know about previous DOS versions.

Using MSD, Microsoft's Diagnostic Software

If you are using DOS 6 or Windows 3.1, you can use the MSD command to learn a great deal about your computer's hardware settings. Specifically, MSD will display information about the following:

CPU type
Video type
Mouse type
Disk types
COM ports
Memory-resident programs
Network settings
Memory use
DOS version
Adapter types
LPT ports
IRQ settings
Device drivers

The format of the MSD command is

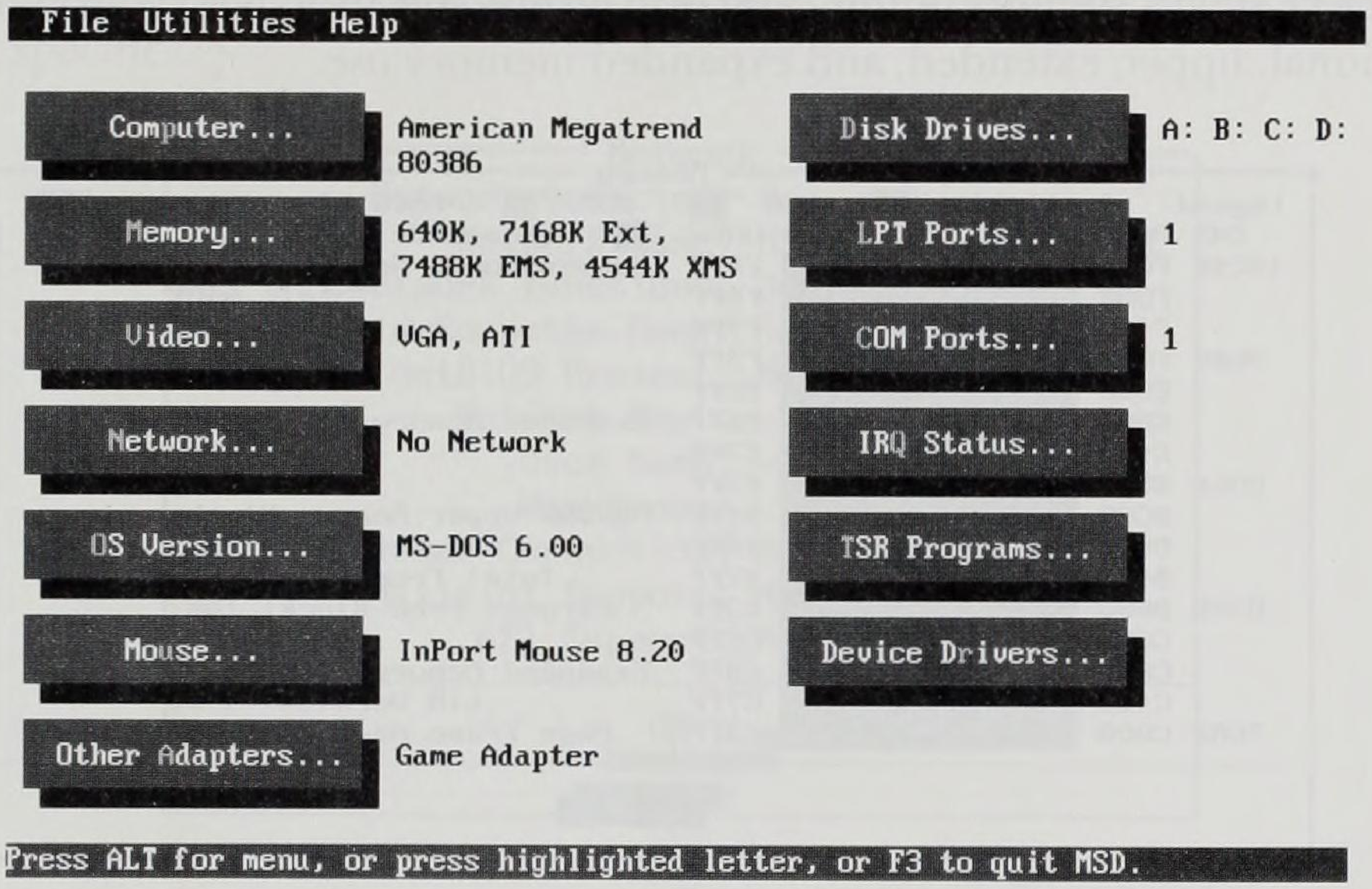
```
MSD [/B][/F:Filename][/I][/P:Filename][/S:Filename]
```

where /B directs MSD to run in black and white; /F:*Filename* directs MSD to create a complete diagnostic report that contains your name, company, and address, as well as MSD's diagnostic findings; /I directs MSD not to perform its initial hardware detection; /P:*Filename* directs MSD to create a complete diagnostic report minus your name and address; and /S:*Filename* directs MSD to create a summary report.

MSD is a menu-driven program. Invoke it from the DOS prompt as follows:

```
C:\> MSD <ENTER>
```

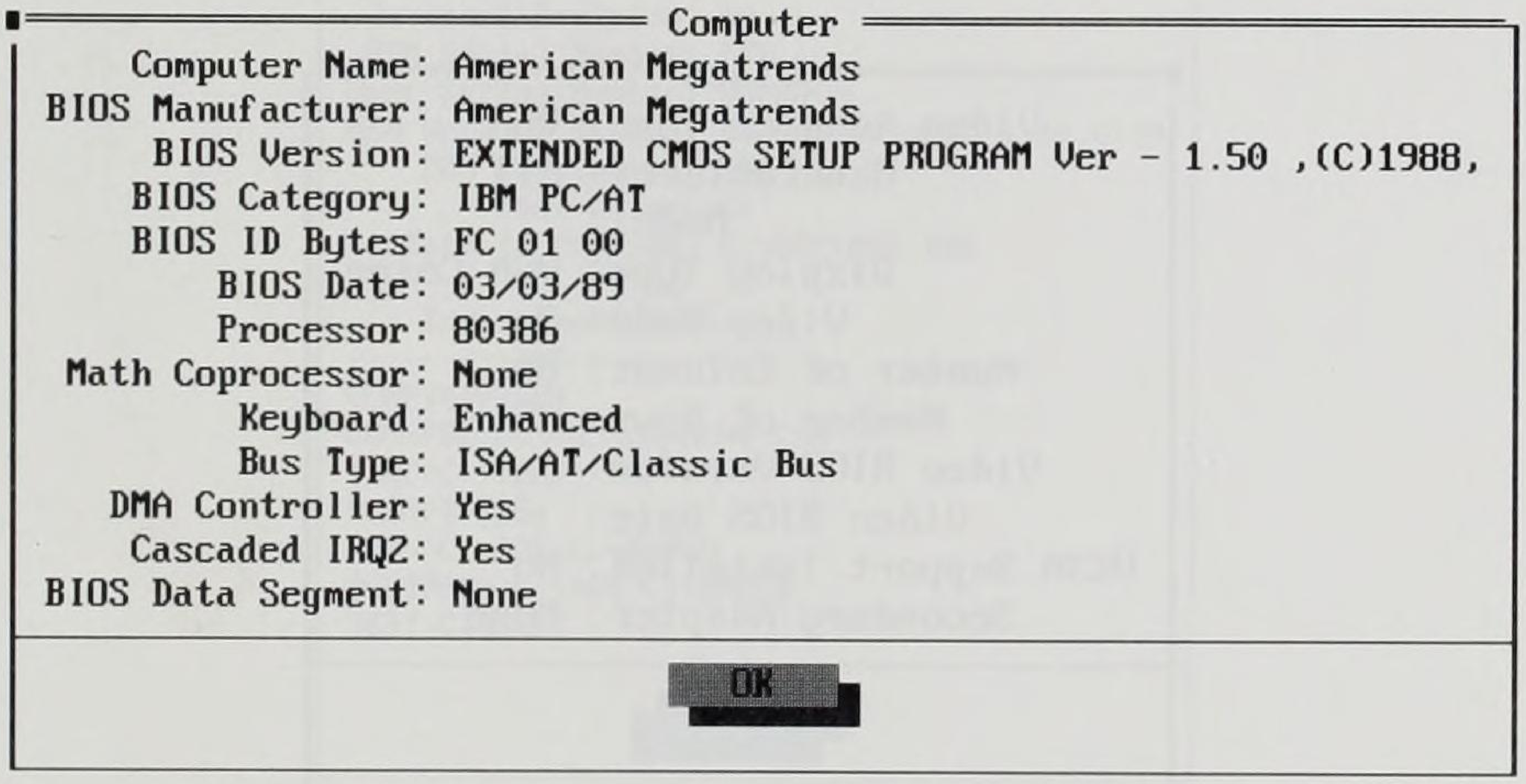
MSD will display its main menu of options, as shown here:



```
 File  Utilities  Help

 Computer...          American Megatrend      Disk Drives...      A: B: C: D:
                      80386
 Memory...            640K, 7168K Ext,        LPT Ports...        1
                      7488K EMS, 4544K XMS
 Video...             VGA, ATI                COM Ports...        1
 Network...           No Network              IRQ Status...
 OS Version...        MS-DOS 6.00             TSR Programs...
 Mouse...             InPort Mouse 8.20       Device Drivers...
 Other Adapters...    Game Adapter

Press ALT for menu, or press highlighted letter, or F3 to quit MSD.
```

Displaying Specifics About Your Computer

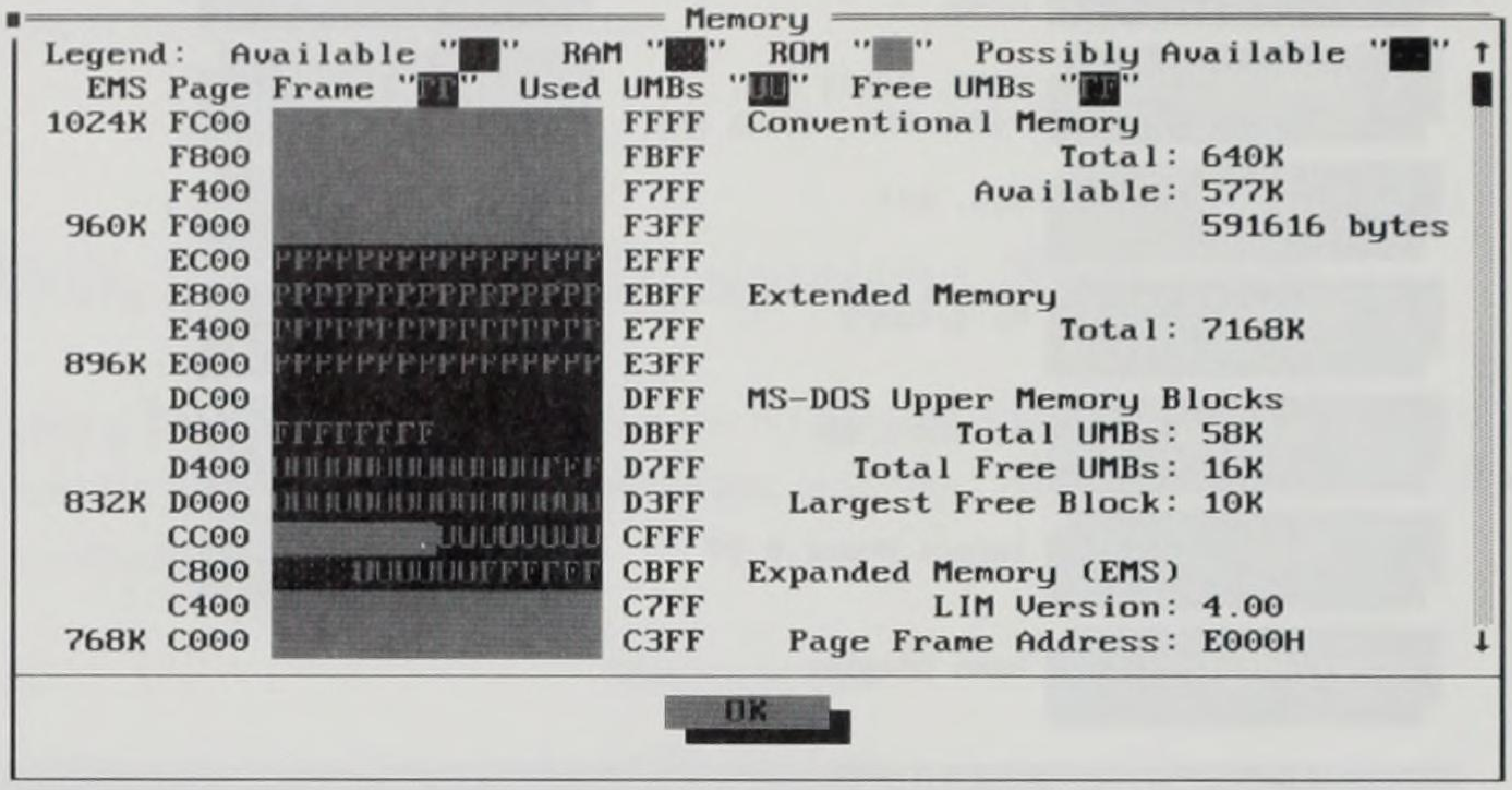
If you select MSD's Computer option, MSD will display information describing your BIOS and CPU type, as well as other computer specifics, as shown here:



```
                         Computer
      Computer Name: American Megatrends
  BIOS Manufacturer: American Megatrends
       BIOS Version: EXTENDED CMOS SETUP PROGRAM Ver - 1.50 ,(C)1988,
      BIOS Category: IBM PC/AT
      BIOS ID Bytes: FC 01 00
          BIOS Date: 03/03/89
          Processor: 80386
   Math Coprocessor: None
           Keyboard: Enhanced
           Bus Type: ISA/AT/Classic Bus
     DMA Controller: Yes
      Cascaded IRQ2: Yes
  BIOS Data Segment: None

                            OK
```

Viewing Your Memory Usage

If your select MSD's Memory option, MSD will display specifics about your computer's conventional, upper, extended, and expanded memory use.

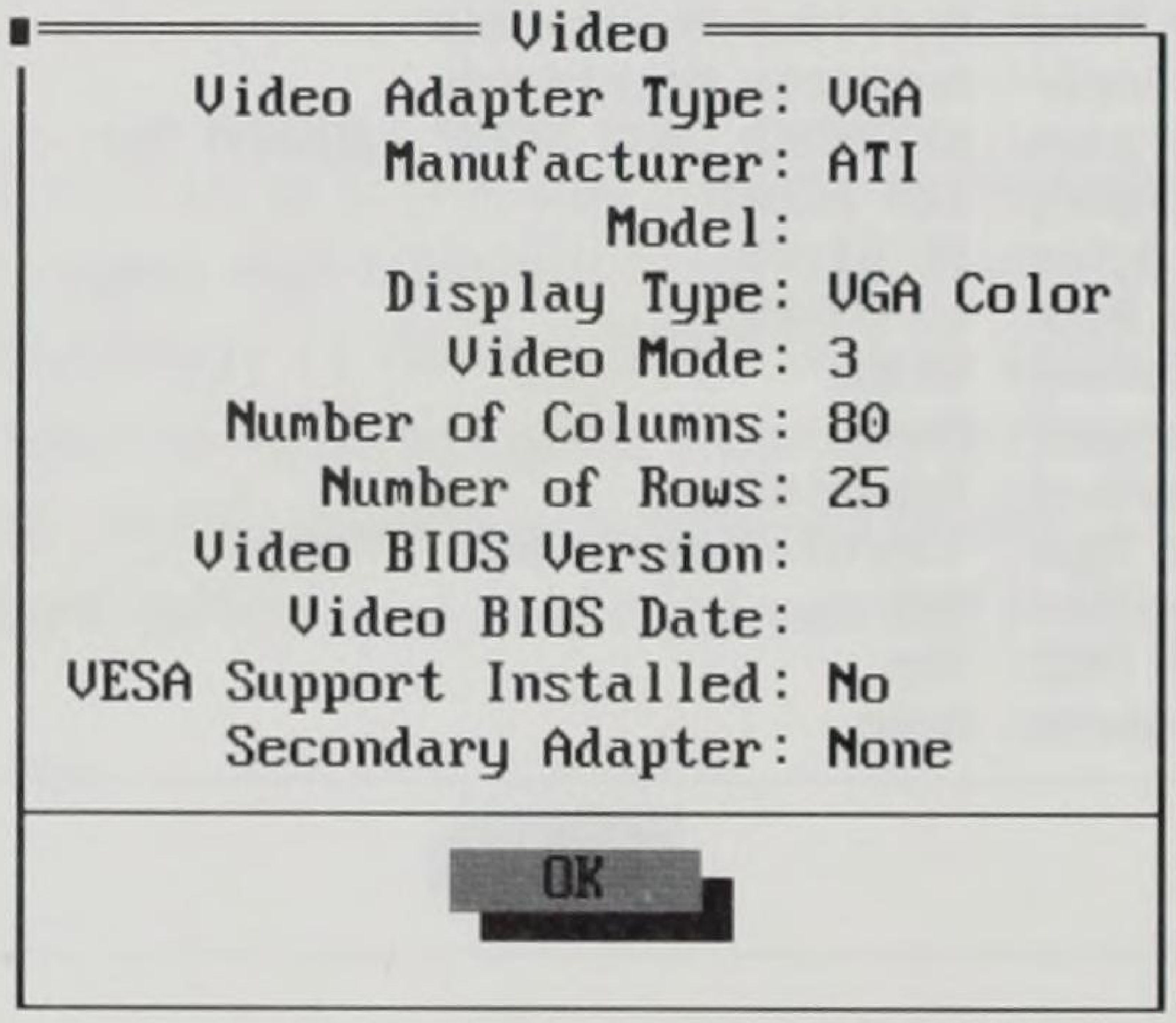


```
                              Memory
Legend:  Available "■■"  RAM "■■"  ROM "■■"  Possibly Available "■■"
  EMS Page Frame "PP"  Used UMBs "UU"  Free UMBs "FF"
1024K FC00                  FFFF  Conventional Memory
      F800                  FBFF                 Total: 640K
      F400                  F7FF             Available: 577K
 960K F000                  F3FF                        591616 bytes
      EC00                  EFFF
      E800                  EBFF  Extended Memory
      E400                  E7FF                 Total: 7168K
 896K E000                  E3FF
      DC00                  DFFF  MS-DOS Upper Memory Blocks
      D800                  DBFF            Total UMBs: 58K
      D400                  D7FF       Total Free UMBs: 16K
 832K D000                  D3FF     Largest Free Block: 10K
      CC00                  CFFF
      C800                  CBFF  Expanded Memory (EMS)
      C400                  C7FF           LIM Version: 4.00
 768K C000                  C3FF     Page Frame Address: E000H
```

If you press your computer's UP ARROW and DOWN ARROW keys, you can direct MSD to display additional specifics about your memory use.

Displaying Video Specifics

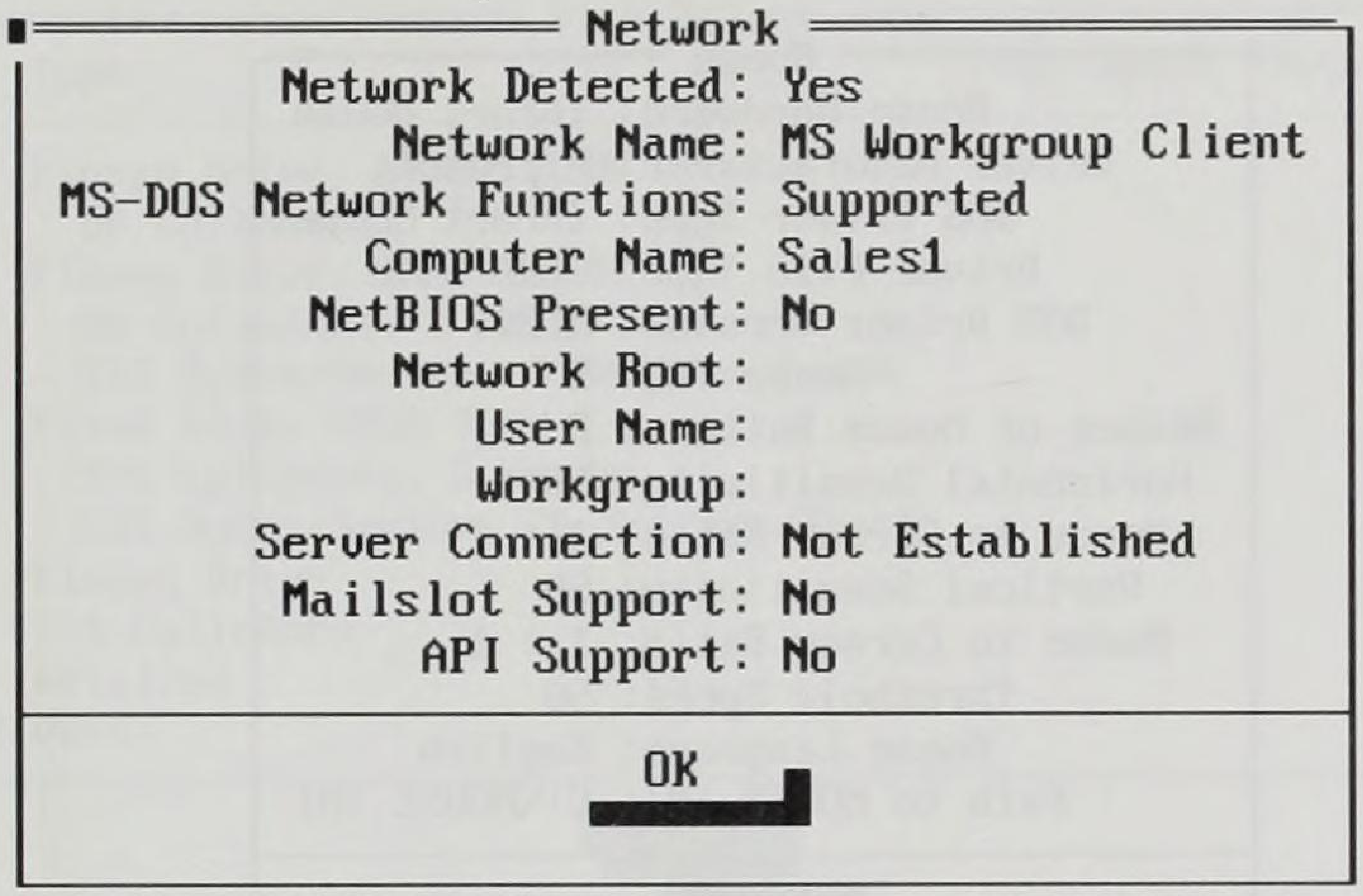
If you select MSD's Video option, MSD will display specifics about your video configuration.



```
                   Video
      Video Adapter Type: VGA
            Manufacturer: ATI
                   Model:
            Display Type: VGA Color
              Video Mode: 3
       Number of Columns: 80
          Number of Rows: 25
      Video BIOS Version:
         Video BIOS Date:
  VESA Support Installed: No
       Secondary Adapter: None
```

Displaying Network Information

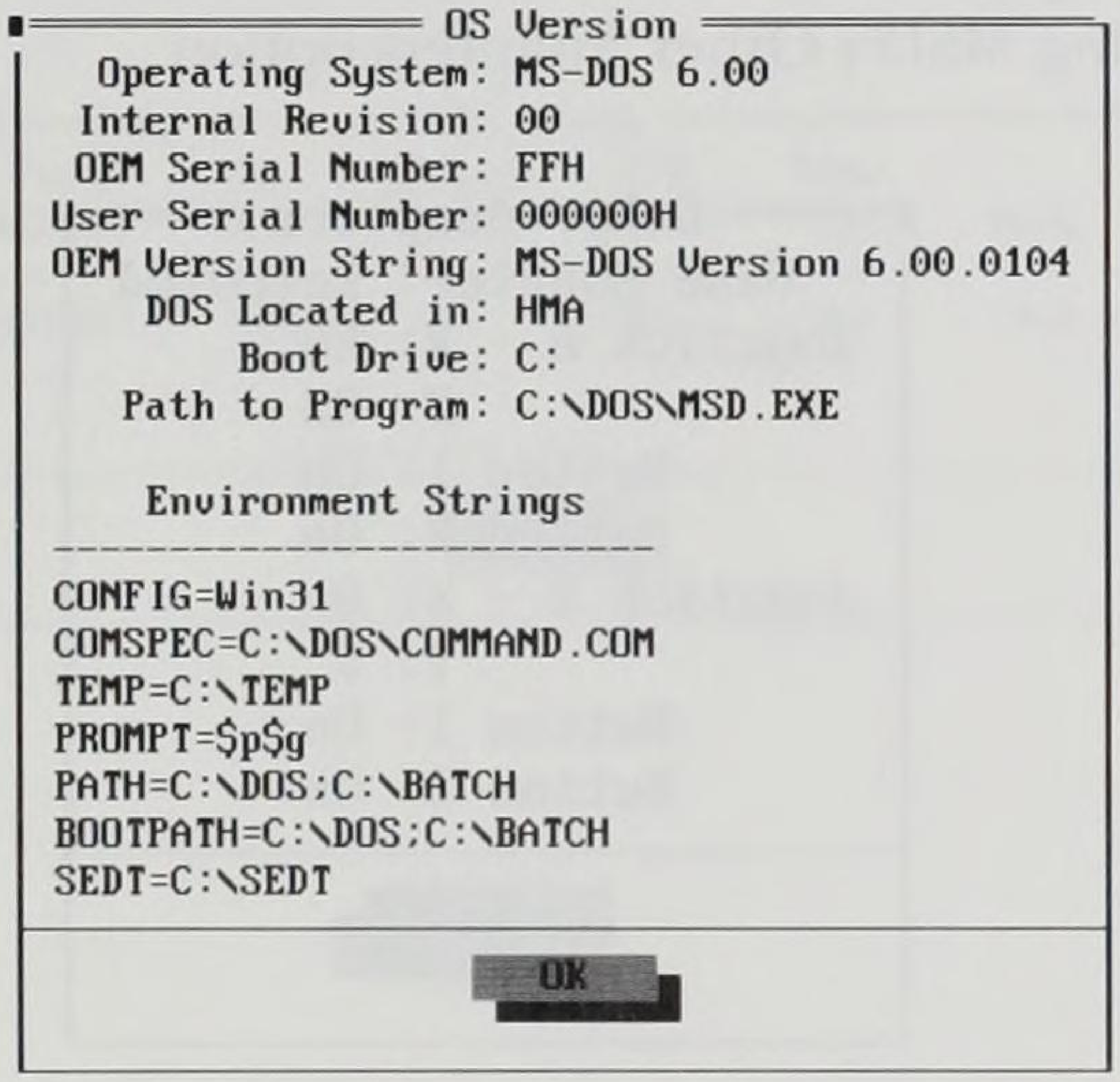
If your computer is connected to a local area network, you can use MSD to display network specifics.



```
       Network Detected: Yes
           Network Name: MS Workgroup Client
MS-DOS Network Functions: Supported
          Computer Name: Sales1
         NetBIOS Present: No
           Network Root:
              User Name:
              Workgroup:
      Server Connection: Not Established
        Mailslot Support: No
            API Support: No
```

Displaying Operating System Specifics

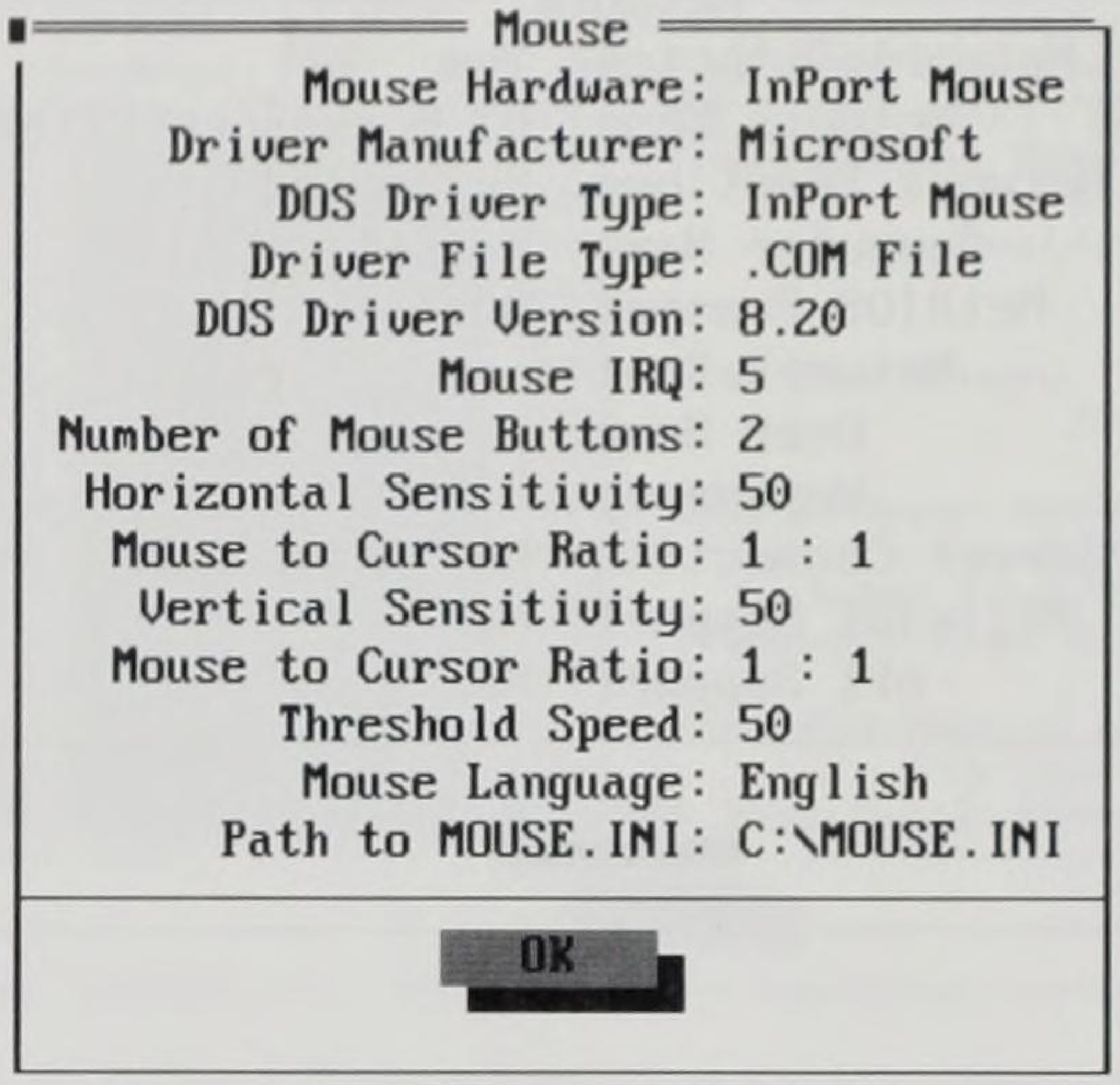
If you select MSD's OS Version option, MSD will display specifics about the current DOS version, as well as the current environment settings.



```
  Operating System: MS-DOS 6.00
 Internal Revision: 00
 OEM Serial Number: FFH
User Serial Number: 000000H
OEM Version String: MS-DOS Version 6.00.0104
    DOS Located in: HMA
        Boot Drive: C:
   Path to Program: C:\DOS\MSD.EXE

    Environment Strings
--------------------------
CONFIG=Win31
COMSPEC=C:\DOS\COMMAND.COM
TEMP=C:\TEMP
PROMPT=$p$g
PATH=C:\DOS;C:\BATCH
BOOTPATH=C:\DOS;C:\BATCH
SEDT=C:\SEDT
```

Displaying Mouse Specifics

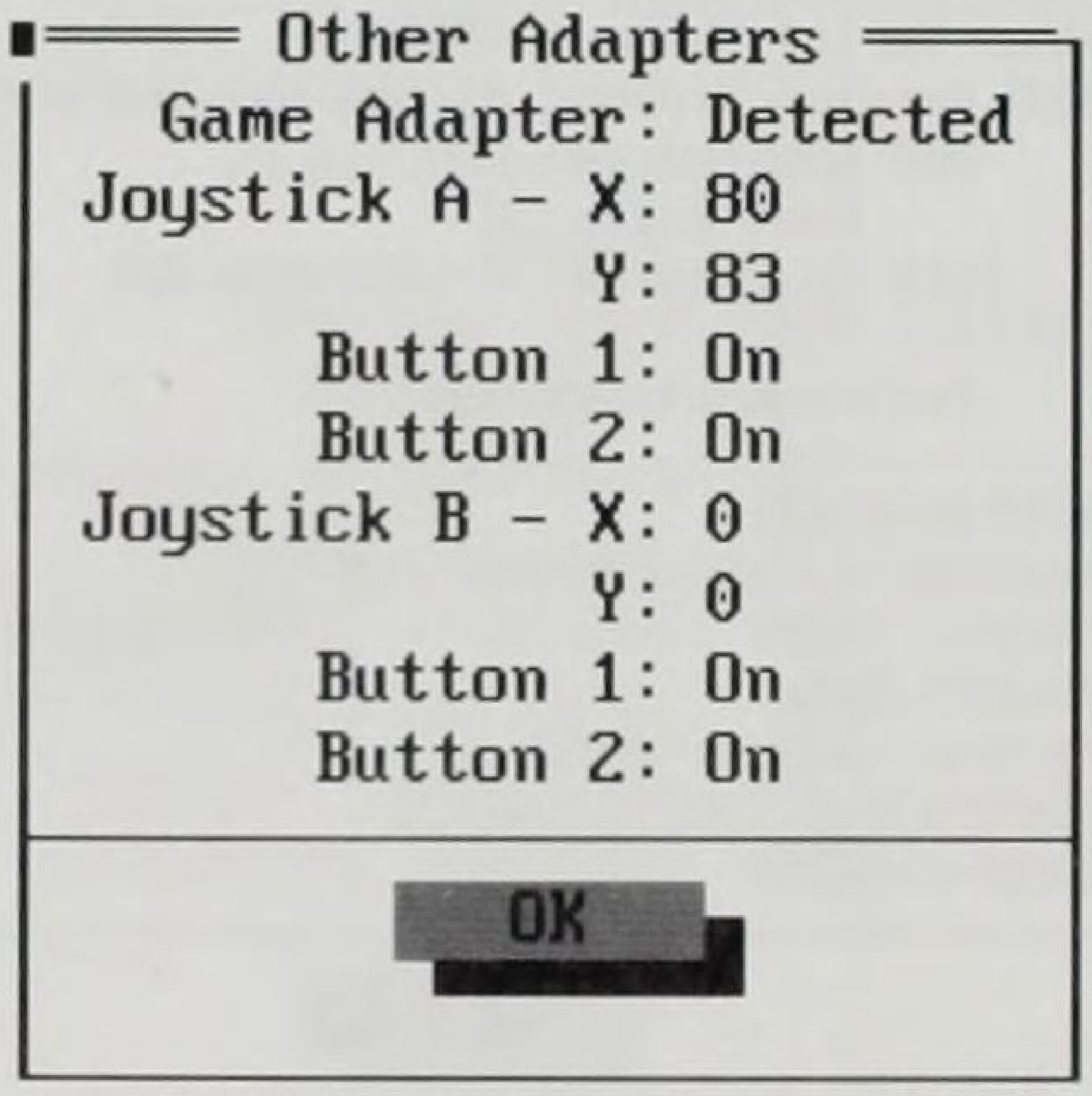
If you select MSD's Mouse settings, MSD will display information about the current mouse driver and various mouse settings.



```
          Mouse Hardware: InPort Mouse
     Driver Manufacturer: Microsoft
         DOS Driver Type: InPort Mouse
        Driver File Type: .COM File
      DOS Driver Version: 8.20
               Mouse IRQ: 5
Number of Mouse Buttons: 2
  Horizontal Sensitivity: 50
   Mouse to Cursor Ratio: 1 : 1
    Vertical Sensitivity: 50
   Mouse to Cursor Ratio: 1 : 1
         Threshold Speed: 50
          Mouse Language: English
        Path to MOUSE.INI: C:\MOUSE.INI
```

Displaying Adapter Information

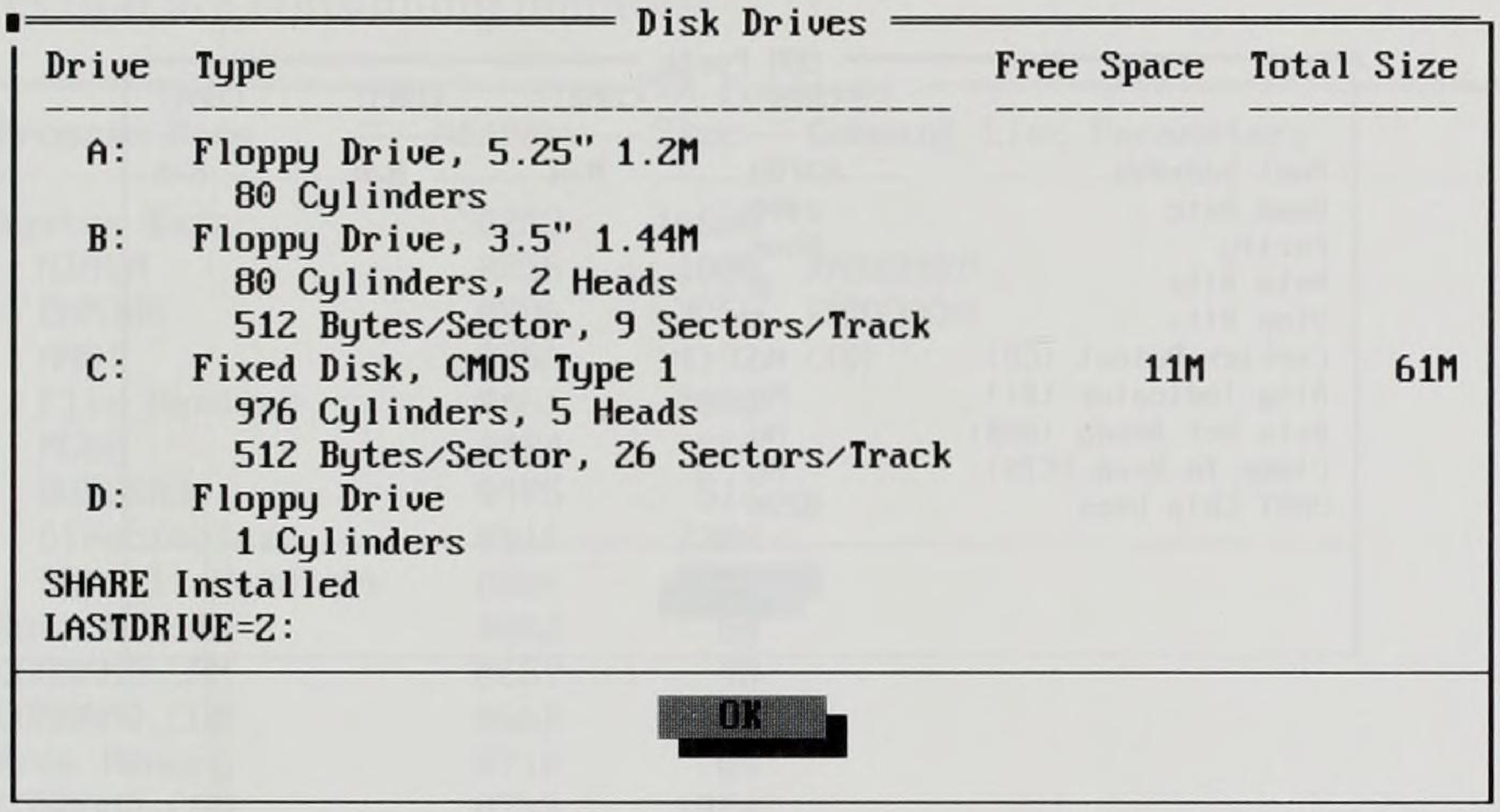
If your computer has a joystick or other device installed, you may be able to display the device settings using MSD's Other Adapters option.



```
  Game Adapter: Detected
Joystick A - X: 80
             Y: 83
      Button 1: On
      Button 2: On
Joystick B - X: 0
             Y: 0
      Button 1: On
      Button 2: On
```

Displaying Disk Drive Specifics

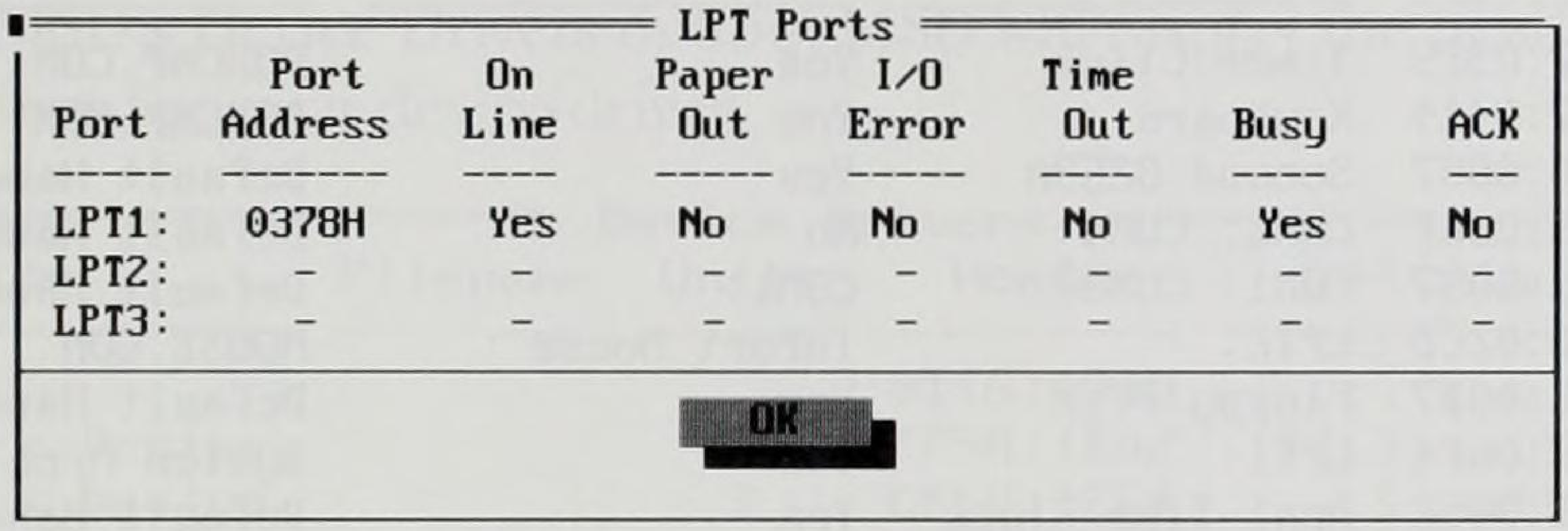
If you select MSD's Disk Drives option, MSD will display information describing each of your disk drives.



```
Disk Drives
Drive Type                                     Free Space  Total Size
----- ---------------------------------------- ----------  ----------
  A:  Floppy Drive, 5.25" 1.2M
        80 Cylinders
  B:  Floppy Drive, 3.5" 1.44M
        80 Cylinders, 2 Heads
        512 Bytes/Sector, 9 Sectors/Track
  C:  Fixed Disk, CMOS Type 1                          11M         61M
        976 Cylinders, 5 Heads
        512 Bytes/Sector, 26 Sectors/Track
  D:  Floppy Drive
        1 Cylinders
SHARE Installed
LASTDRIVE=Z:
```

Displaying LPT Port Status Information

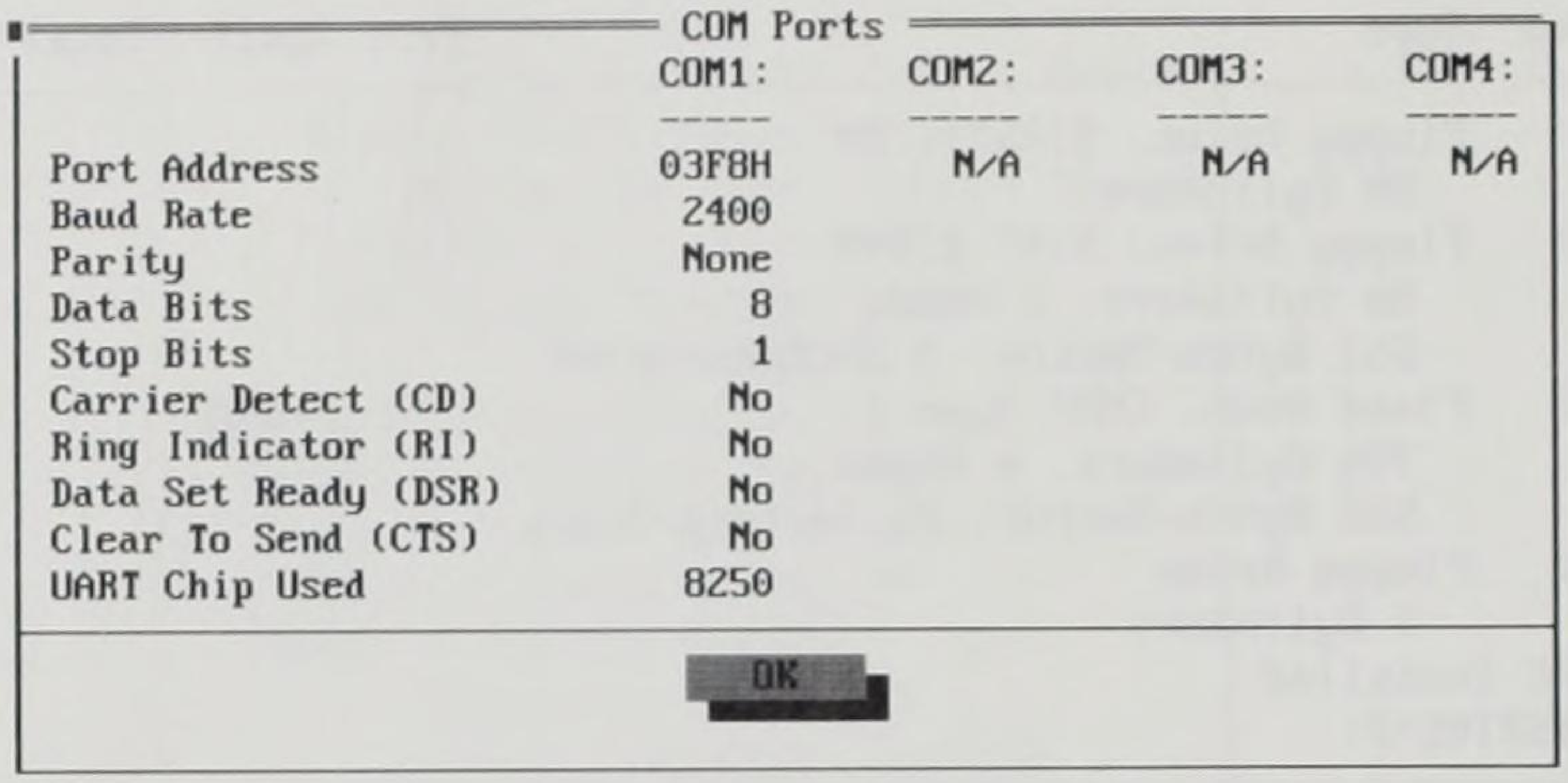
If you select MSD's LPT Ports option, MSD will display information describing each of your computer's parallel ports.



```
LPT Ports
         Port     On    Paper   I/O    Time
Port    Address  Line    Out   Error   Out    Busy   ACK
-----   -------  ----   -----  -----   ----   ----   ---
LPT1:    0378H    Yes    No     No      No     Yes    No
LPT2:      -       -      -      -       -      -      -
LPT3:      -       -      -      -       -      -      -
```

Displaying COM Port Status Information

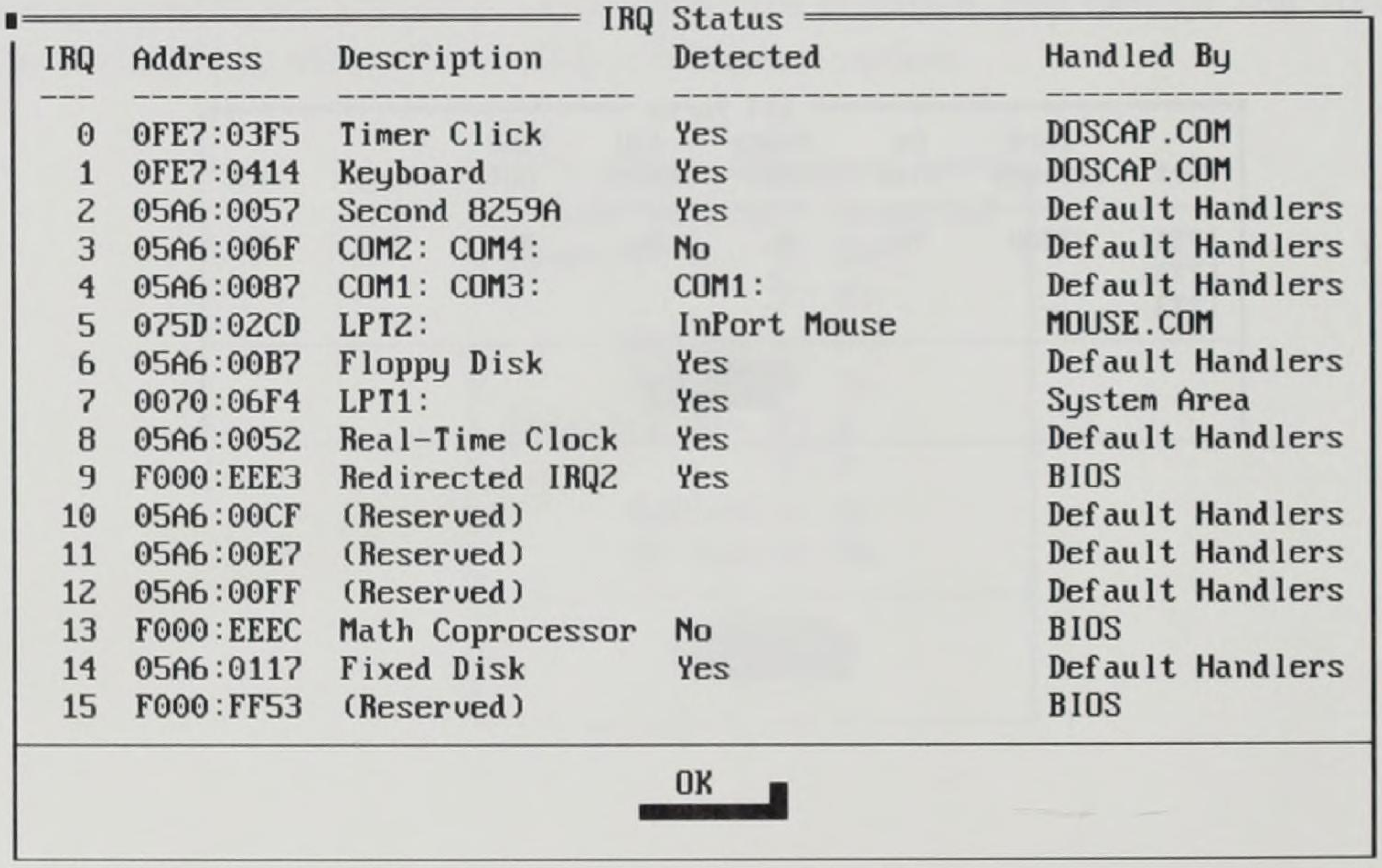
If you select MSD's COM Ports option, MSD will display specifics about your serial ports such as their baud rate and other data communication settings.



```
                              COM Ports
                              COM1:      COM2:      COM3:      COM4:
                              -----      -----      -----      -----
Port Address                  03F8H        N/A        N/A        N/A
Baud Rate                      2400
Parity                         None
Data Bits                         8
Stop Bits                         1
Carrier Detect (CD)              No
Ring Indicator (RI)              No
Data Set Ready (DSR)             No
Clear To Send (CTS)              No
UART Chip Used                 8250
```

Displaying IRQ Settings

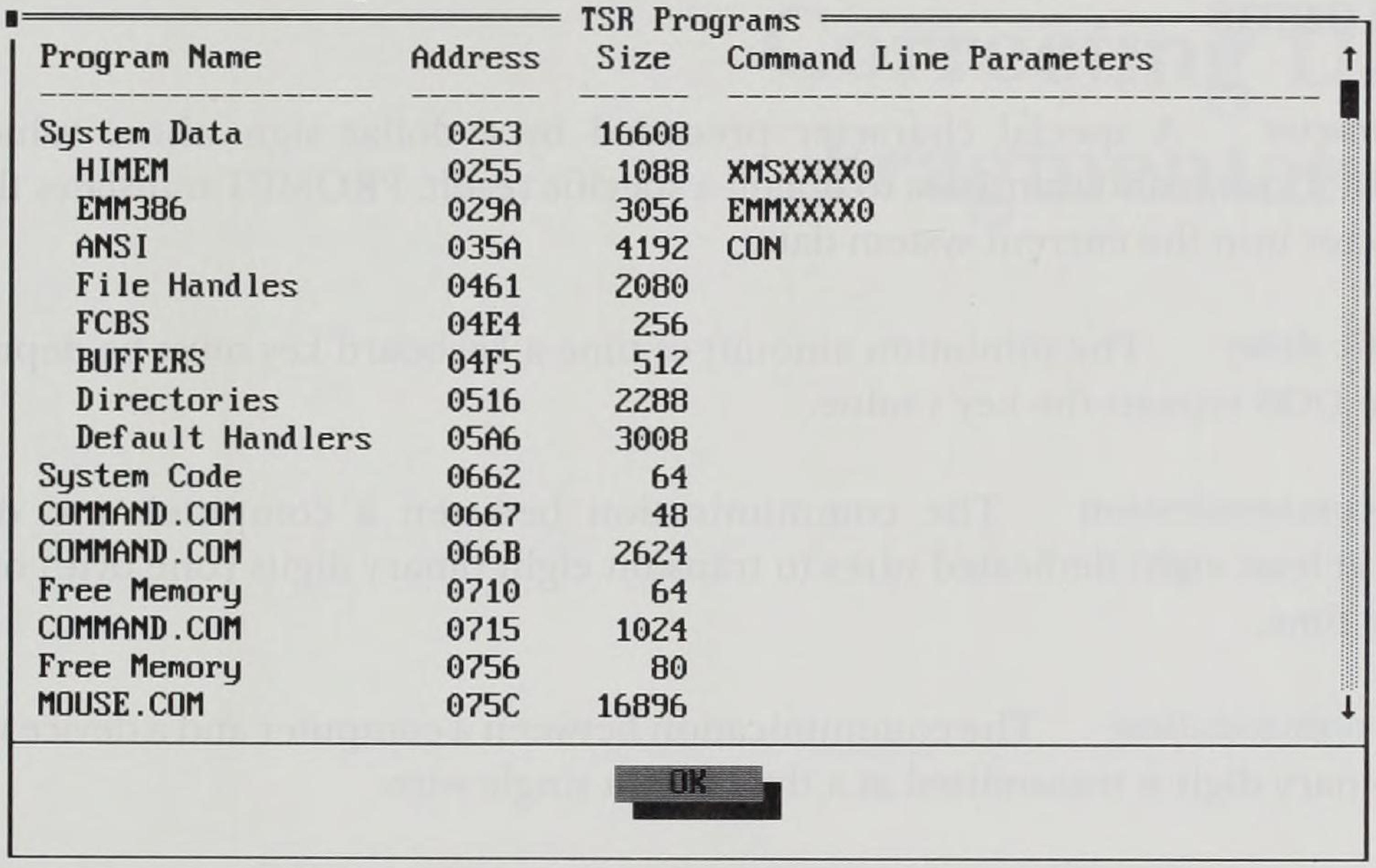
IRQ is an abbreviation for *interrupt request.* When you install a hardware board inside your computer, each board uses a unique wire to interrupt the processor. That line is called an interrupt request (IRQ) line. In most cases, you must assign a unique IRQ setting to each device. If you select MSD's IRQ Status option, MSD will display the current IRQ settings.



```
                              IRQ Status
IRQ  Address    Description       Detected            Handled By
---  ---------  ----------------  ------------------  ----------------
  0  0FE7:03F5  Timer Click       Yes                 DOSCAP.COM
  1  0FE7:0414  Keyboard          Yes                 DOSCAP.COM
  2  05A6:0057  Second 8259A      Yes                 Default Handlers
  3  05A6:006F  COM2: COM4:       No                  Default Handlers
  4  05A6:0087  COM1: COM3:       COM1:               Default Handlers
  5  075D:02CD  LPT2:             InPort Mouse        MOUSE.COM
  6  05A6:00B7  Floppy Disk       Yes                 Default Handlers
  7  0070:06F4  LPT1:             Yes                 System Area
  8  05A6:0052  Real-Time Clock   Yes                 Default Handlers
  9  F000:EEE3  Redirected IRQ2   Yes                 BIOS
 10  05A6:00CF  (Reserved)                            Default Handlers
 11  05A6:00E7  (Reserved)                            Default Handlers
 12  05A6:00FF  (Reserved)                            Default Handlers
 13  F000:EEEC  Math Coprocessor  No                  BIOS
 14  05A6:0117  Fixed Disk        Yes                 Default Handlers
 15  F000:FF53  (Reserved)                            BIOS
```

Displaying Memory-Resident Programs

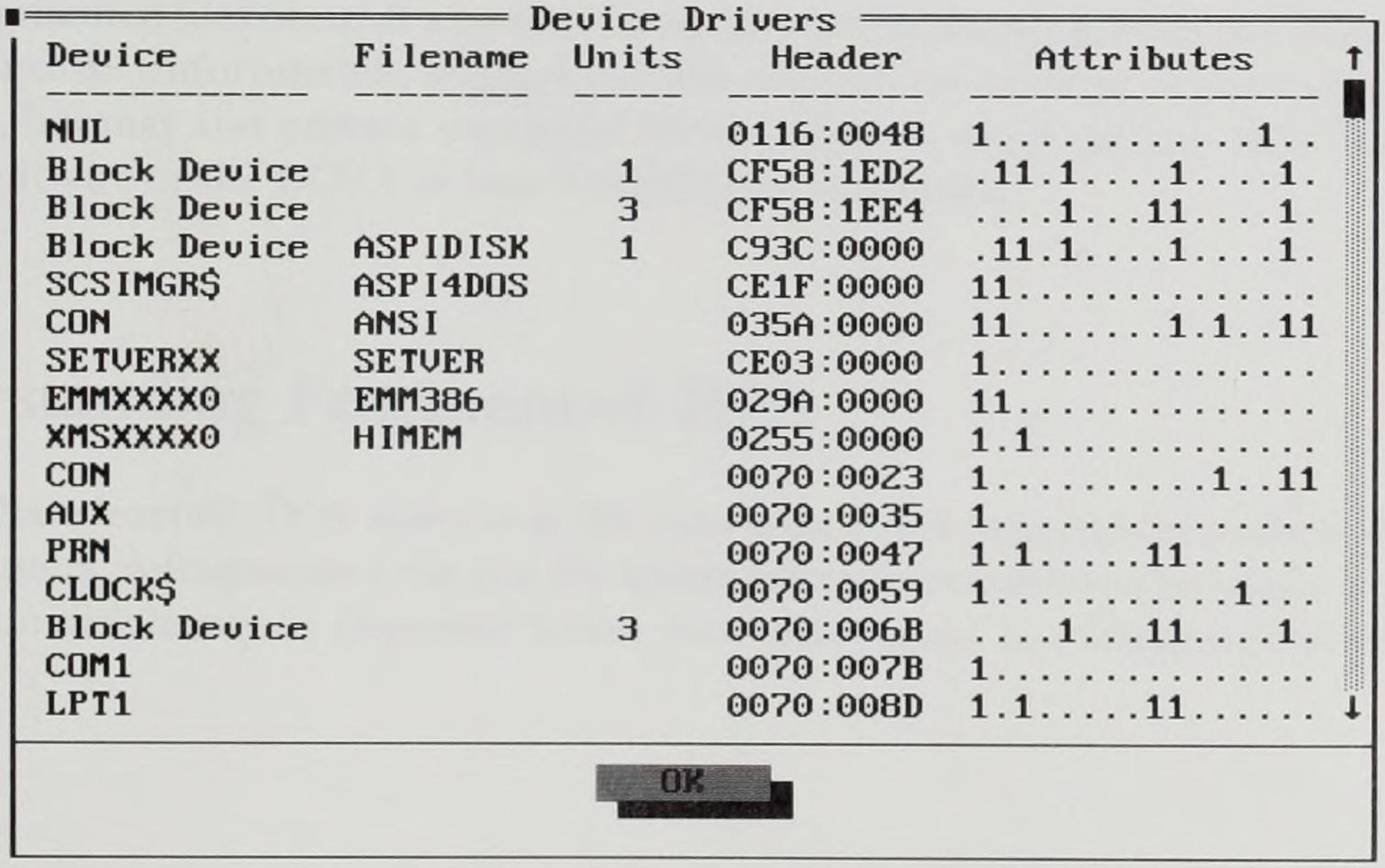
If you select MSD's TSR (terminate-and-stay-resident) Programs option, MSD will display the starting address and size of memory-resident programs and DOS data structures that are consuming memory.



```
═══════════════════════ TSR Programs ═══════════════════════
Program Name        Address  Size   Command Line Parameters
------------------  -------  ------ -------------------------------
System Data          0253    16608
  HIMEM              0255     1088  XMSXXXX0
  EMM386             029A     3056  EMMXXXX0
  ANSI               035A     4192  CON
  File Handles       0461     2080
  FCBS               04E4      256
  BUFFERS            04F5      512
  Directories        0516     2288
  Default Handlers   05A6     3008
System Code          0662       64
COMMAND.COM          0667       48
COMMAND.COM          066B     2624
Free Memory          0710       64
COMMAND.COM          0715     1024
Free Memory          0756       80
MOUSE.COM            075C    16896
```

Displaying Device Driver Specifics

If you select MSD's Device Drivers option, MSD will display the header address and attribute settings for each device driver.



```
════════════════════ Device Drivers ════════════════════
Device        Filename  Units   Header       Attributes
------------  --------  -----  ---------  ----------------
NUL                            0116:0048  1............1..
Block Device              1    CF58:1ED2  .11.1....1....1.
Block Device              3    CF58:1EE4  ....1...11....1.
Block Device  ASPIDISK    1    C93C:0000  .11.1....1....1.
SCSIMGR$      ASPI4DOS         CE1F:0000  11..............
CON           ANSI             035A:0000  11.......1.1..11
SETVERXX      SETVER           CE03:0000  1...............
EMMXXXX0      EMM386           029A:0000  11..............
XMSXXXX0      HIMEM            0255:0000  1.1.............
CON                            0070:0023  1..........1..11
AUX                            0070:0035  1...............
PRN                            0070:0047  1.1.....11......
CLOCK$                         0070:0059  1...........1...
Block Device              3    0070:006B  ....1...11....1.
COM1                           0070:007B  1...............
LPT1                           0070:008D  1.1.....11......
```

For a detailed explanation of DOS device drivers and device attributes, see the book *DOS Programming: The Complete Reference* (Osborne/McGraw-Hill).

Key Terms

Metacharacter A special character preceded by a dollar sign whose value the PROMPT command translates to obtain a specific result. PROMPT translates the $D character into the current system date.

Typematic delay The minimum amount of time a keyboard key must be depressed before DOS repeats the key's value.

Parallel communication The communication between a computer and device using at least eight dedicated wires to transmit eight binary digits (one byte) of data at one time.

Serial communication The communication between a computer and a device where one binary digit is transmitted at a time over a single wire.

Baud rate The communication speed between a computer and a device, typically measured in bits per second.

Chapter 42

Detecting and Correcting Disk Fragmentation

If you are like most DOS users, it won't be long before you look for ways to make your system run faster. In Chapter 31 you read about several DOS performance improvement techniques. Briefly mentioned in Chapter 31 were the identification and correction of fragmented files. As you will learn in this chapter, a *fragmented* file is a file whose contents are stored in sectors dispersed across your disk. If your programs are taking longer to start, or your word processing or spreadsheet files are taking longer to load, the files on your disk may be fragmented.

Fragmented files occur as a natural result of creating, deleting, and enlarging files on your disk. Unfortunately, fragmented files not only decrease your system performance, but may also prevent successful file undeletions when you use either third-party software or the DOS 5 or later UNDELETE command.

Understanding Fragmented Files

As you have learned, DOS stores your file contents on disk in groups of sectors called disk clusters. A fragmented file is a file whose contents require several clusters that reside on disk locations dispersed across your disk. Figure 42-1 illustrates two files.

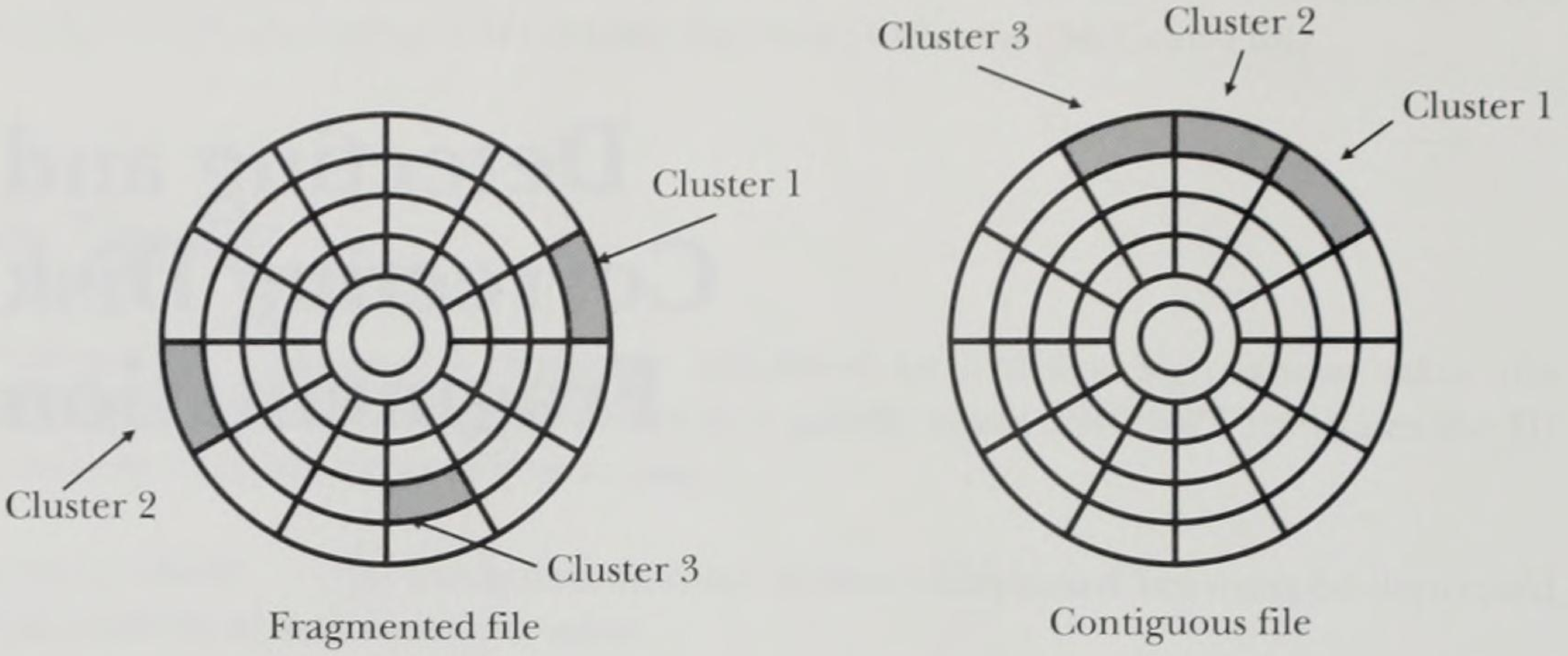
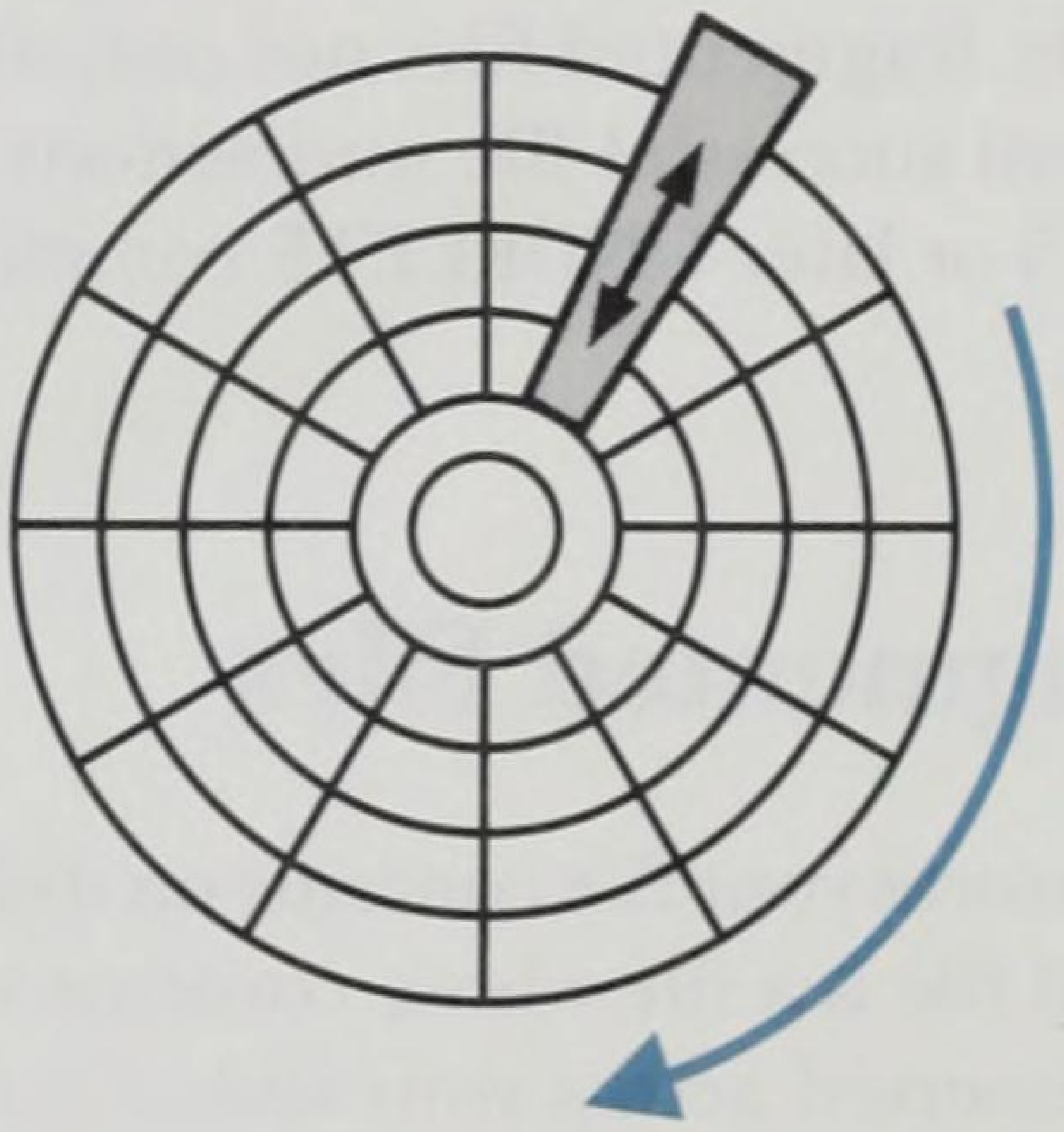


Figure 42-1. *A fragmented file and a contiguous file*

The first file's contents are fragmented. The second file's contents are contiguous, meaning its clusters reside in consecutive storage locations.

Here's how fragmented files decrease your system performance. Your disk drive has a read and write head that reads the information stored on the disk or records new information as required. The disk drive can move the read/write head in and out to access different tracks on the disk. To access different disk sectors, the disk drive must rotate the disk past the read/write head. The following illustration demonstrates how the disk read/write head accesses the information stored on your disk by moving to the correct track and waiting for the desired disk sector to rotate past:



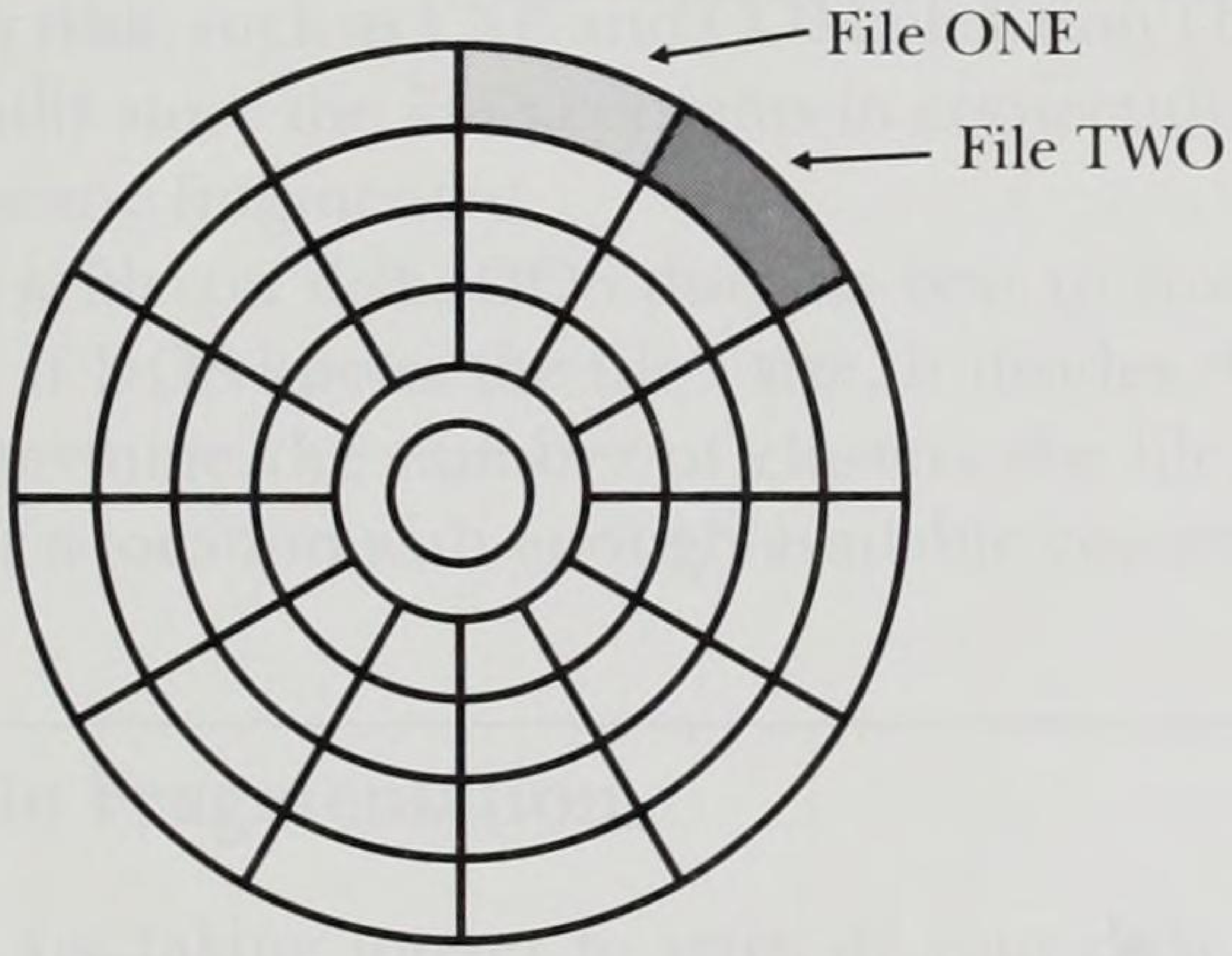
Unlike your computer's fast electronic memory, your disk drive is mechanical, meaning it physically moves. Compared to the speed of electricity, even the smallest physical movement is very slow. To improve your system performance, you need to reduce the amount of this physical disk head movement. Fragmented files, unfortunately, increase the amount of head movement and rotational delay.

Using the files in Figure 42-1, watch the steps DOS must perform to read each file's contents. To begin, the disk drive moves the read/write head to the outermost track. When cluster 1 spins by the read/write head, the drive reads it. Next, using the contiguous file, the disk drive can immediately read the second cluster. For the fragmented file, however, the disk drive must wait for the disk to spin one half a revolution before it can read the second cluster, as illustrated in Figure 42-2.

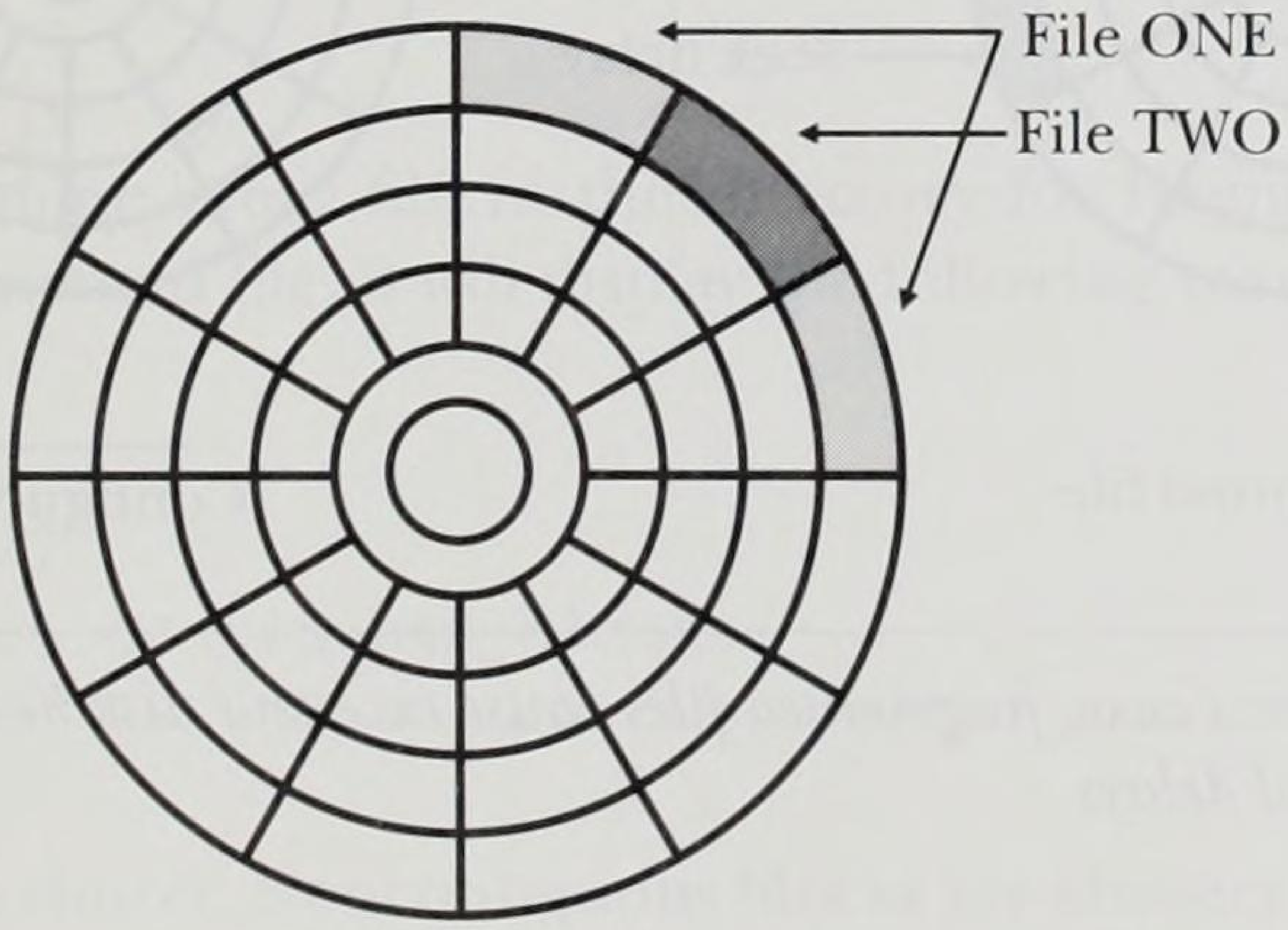
To read cluster 3 of the contiguous file, the disk drive simply reads the next cluster. For the fragmented file, however, the disk drive must first move the read/write head to the correct track, and then wait for the cluster to spin by (see Figure 42-3).

What Causes Fragmented Files?

Fragmented files are a natural occurrence of using your disk to store files. Assume, for example, your disk contains two small files, ONE and TWO, as shown here:



Assume you edit file ONE and increase its size by one cluster. Because file TWO is physically next to file ONE, DOS must use the next available cluster. As a result, file ONE becomes fragmented, as shown here:



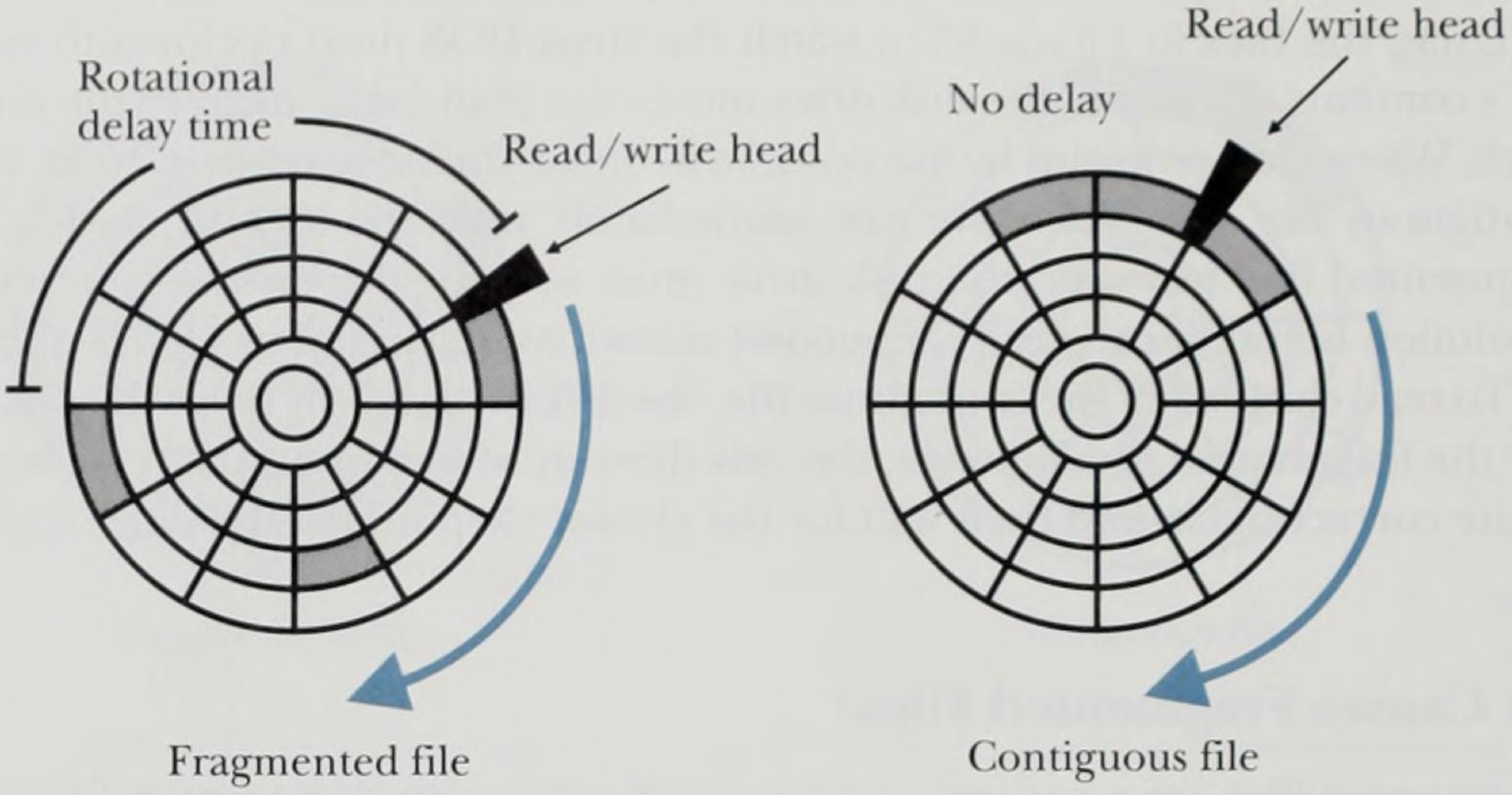

Rotational delay time
Read/write head
Fragmented file
Read/write head
No delay
Contiguous file

Figure 42-2. *Fragmented files cause rotational delays*

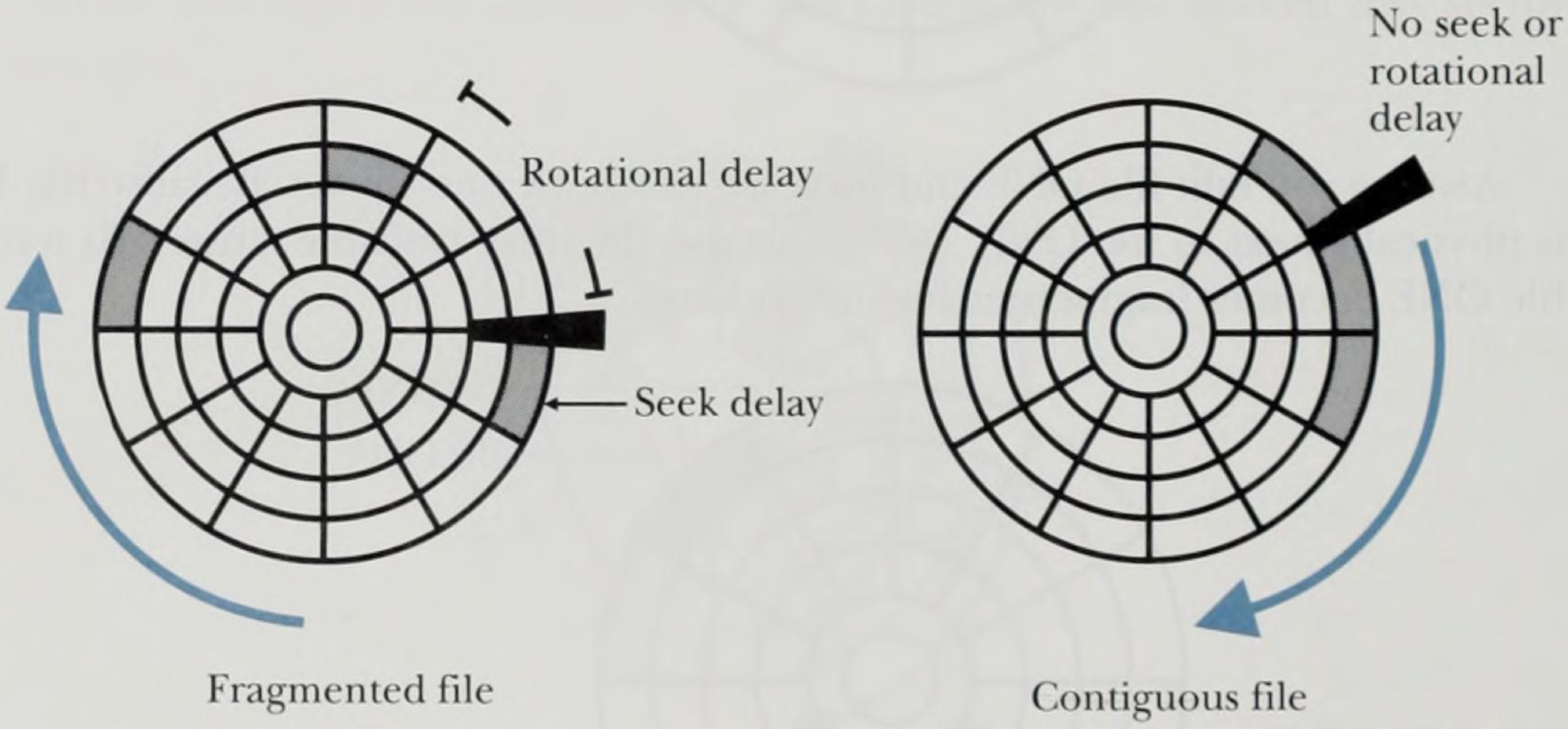
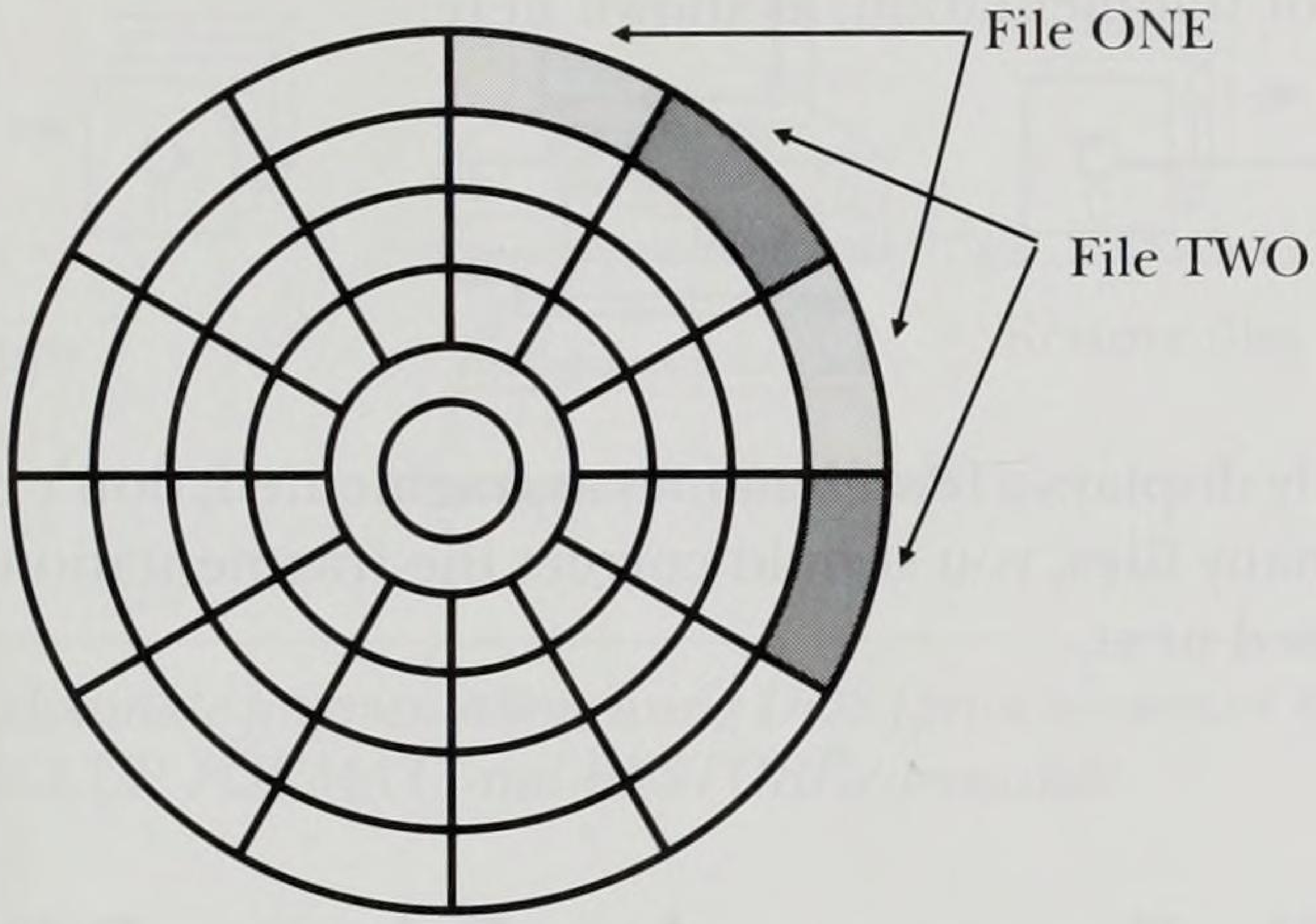

Rotational delay
Seek delay
Fragmented file
No seek or rotational delay
Contiguous file

Figure 42-3. *In the worst case, fragmented files cause excessive disk head movement and rotational delays*

If then later you increase the size of file TWO by one cluster, it will also become fragmented.



As you can see, fragmented files can occur quite easily. Fortunately, however, many of the files on your disk, such as EXE and COM files, don't change in size. Therefore, if DOS can originally store the file's contents in consecutive storage locations, those files will never become fragmented.

When placing a file on disk, DOS does its best to store the file in consecutive storage locations. If DOS knows the file's size, it divides the file's size by the disk's cluster size to determine the number of clusters the file requires. DOS can then search the disk for a location with enough available consecutive clusters.

How to Detect File Fragmentation

If your programs are taking longer to start, or your data files take longer to load, check the directory for fragmented files using CHKDSK as shown here:

```
C:\> CHKDSK  *.* <ENTER>
```

CHKDSK will examine every file in the directory for fragmentation. If CHKDSK encounters a fragmented file, it will display the following message:

```
C:\FILENAME.EXT
Contains nn non-contiguous blocks
```

A block is simply a cluster. Noncontiguous blocks are clusters stored in nonconsecutive locations.

CHKDSK only examines one directory of files at a time for fragmentation. To examine your entire disk, you must invoke CHKDSK individually for each directory or use a third-party disk utility. If you have a large file, you can use CHKDSK to examine the file for fragmentation, as shown here:

```
C:\> CHKDSK FILENAME.EXT <ENTER>
```

If CHKDSK only displays a few filenames as fragmented, don't be too concerned. If CHKDSK lists many files, you should correct the fragmentation using one of the techniques discussed next.

Correcting File Fragmentation Prior to DOS 6

Depending on the number of fragmented files your disk contains, there are two ways to correct the fragmentation. First, when DOS copies a file to your disk, it tries to store the file in consecutive storage locations. If you are trying to eliminate fragmentation for a single file, you can first try copying the file to a different disk, deleting the file from the original, and then copying the file back from the second disk to the original disk. If DOS has enough consecutive clusters, it will use them, eliminating the fragmentation. If your disk is very full, however, the file's fragmentation can actually get worse when you copy the file back to your disk.

If your disk contains many fragmented files, or a file copy operation did not correct the fragmentation, you must use a third-party disk utility program or the BACKUP, FORMAT, and RESTORE commands discussed here to eliminate fragmentation.

To eliminate fragmentation using DOS, you must copy all the files on your hard disk to floppy, format your disk, and then copy the files back to your hard disk, as illustrated in Figure 42-4.

To begin, use the BACKUP command to perform a complete disk backup operation, as discussed in Chapter 22.

```
C:\> BACKUP  C:\*.*  A:  /S <ENTER>
```

WARNING

If the backup operation does not successfully back up every file on your disk, do not continue! Instead, repeat the backup operation until it successfully completes.

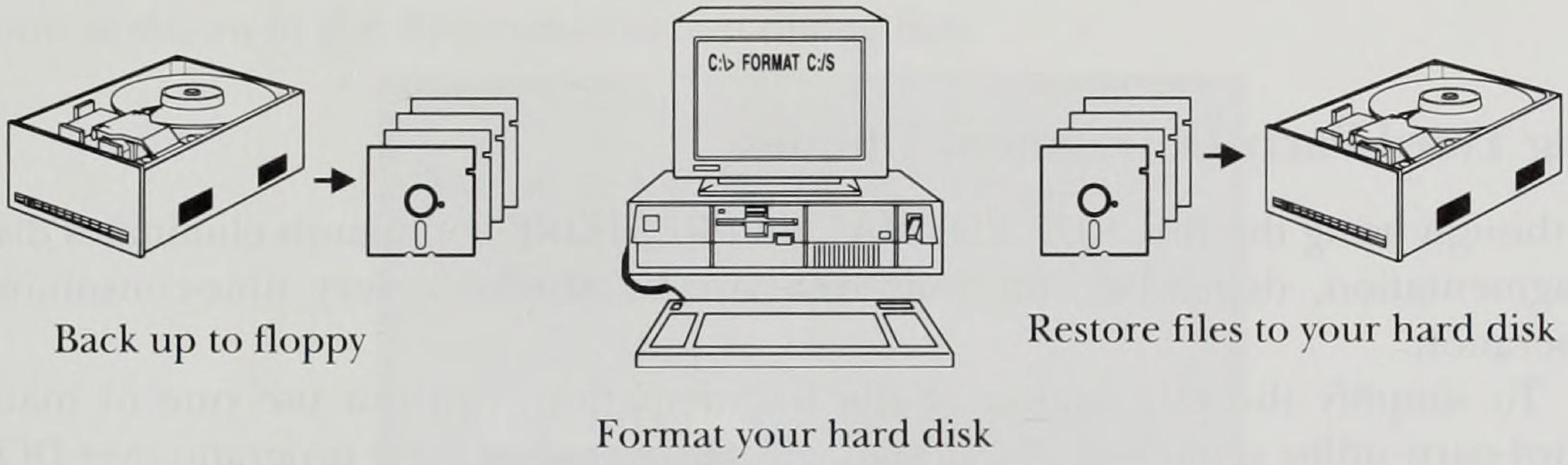

C:\> FORMAT C:/S
Back up to floppy
Restore files to your hard disk
Format your hard disk

Figure 42-4. *To eliminate fragmentation using DOS (prior to version 6), you must use the BACKUP, FORMAT, and RESTORE commands*

After you back up the files on your disk, format a bootable floppy disk.

```
C:\> FORMAT  A:  /S <ENTER>
```

If the disk does not contain COMMAND.COM, copy COMMAND.COM to the disk.

Next, copy the RESTORE command to the floppy disk.

```
C:\> COPY  \DOS\RESTORE.*  A: <ENTER>
```

Label the floppy disk RESTORE and place the floppy disk in a safe location. Next, issue the following command to reformat your hard disk.

```
C:\> FORMAT C:  /S <ENTER>
```

When the format operation completes, place the floppy disk labeled RESTORE in drive A and issue the following RESTORE command, as discussed in Chapter 22.

```
A> RESTORE  A:  C:\*.*  /S <ENTER>
```

After RESTORE places the files back onto your hard disk, the fragmentation is eliminated.

Using Third-Party Defragment Utilities

Although using the BACKUP, FORMAT, and RESTORE commands eliminates disk fragmentation, depending on your disk size, it can be a very time-consuming operation.

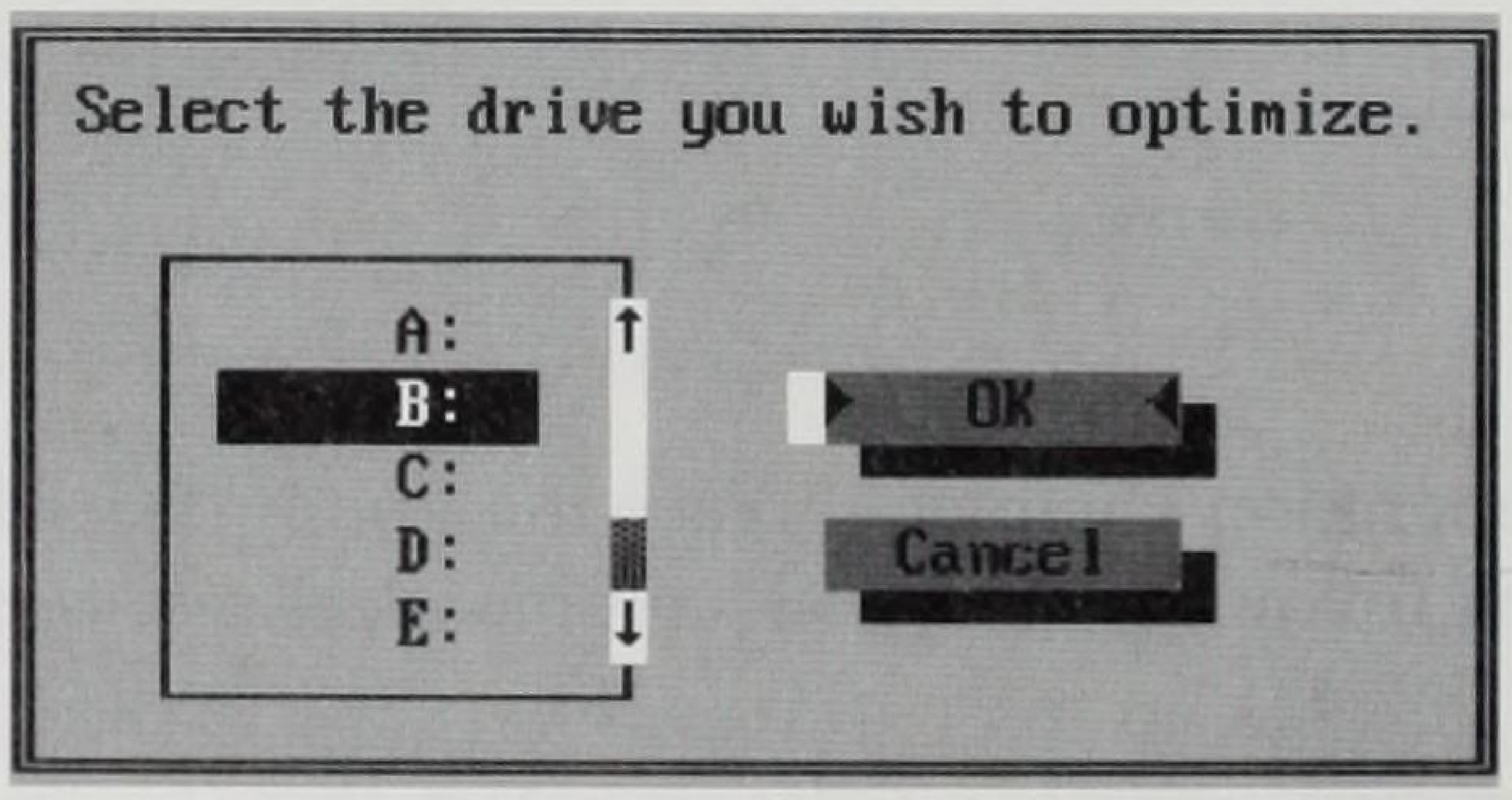
To simplify the elimination of file fragmentation, you can use one of many third-party utility programs. The advantage of using one of these programs over DOS is the utility programs don't need to copy files to floppies and can run on their own without your interaction. Depending on your disk size and number of files, these programs may take an hour or more to eliminate fragmentation. Because the programs can run unattended, however, you can run them at night before you go home. When you return to work, your files will be contiguous.

Defragmenting Files Using DOS 6

If you are using DOS 6, you can quickly defragment the files on your disk using the DEFRAG command. To defragment your files, invoke DEFRAG from the DOS prompt as shown here:

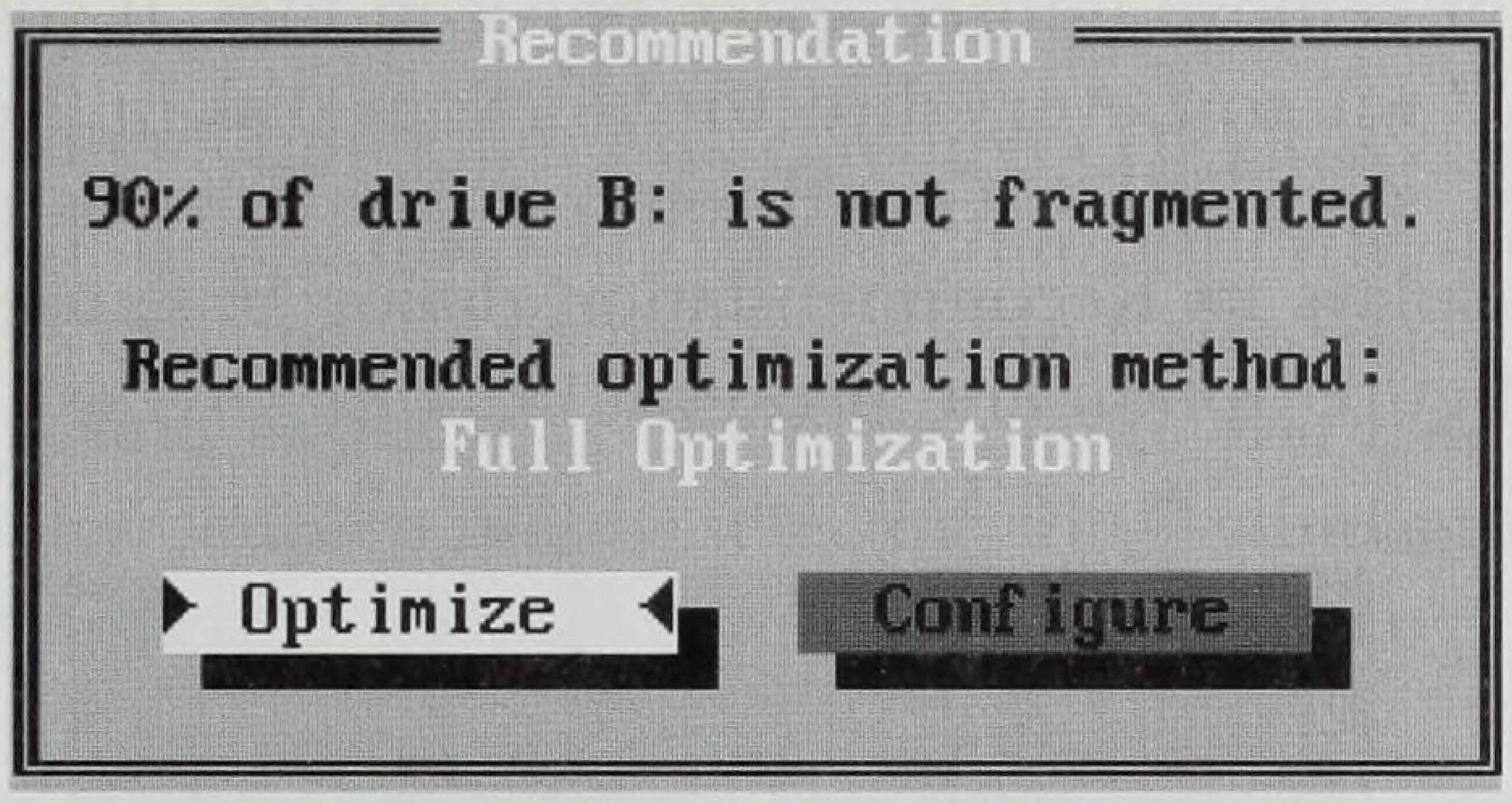
```
C:\> DEFRAG <ENTER>
```

DEFRAG will display a dialog box asking you to select the desired disk drive as shown here:



Select the drive you wish to optimize.

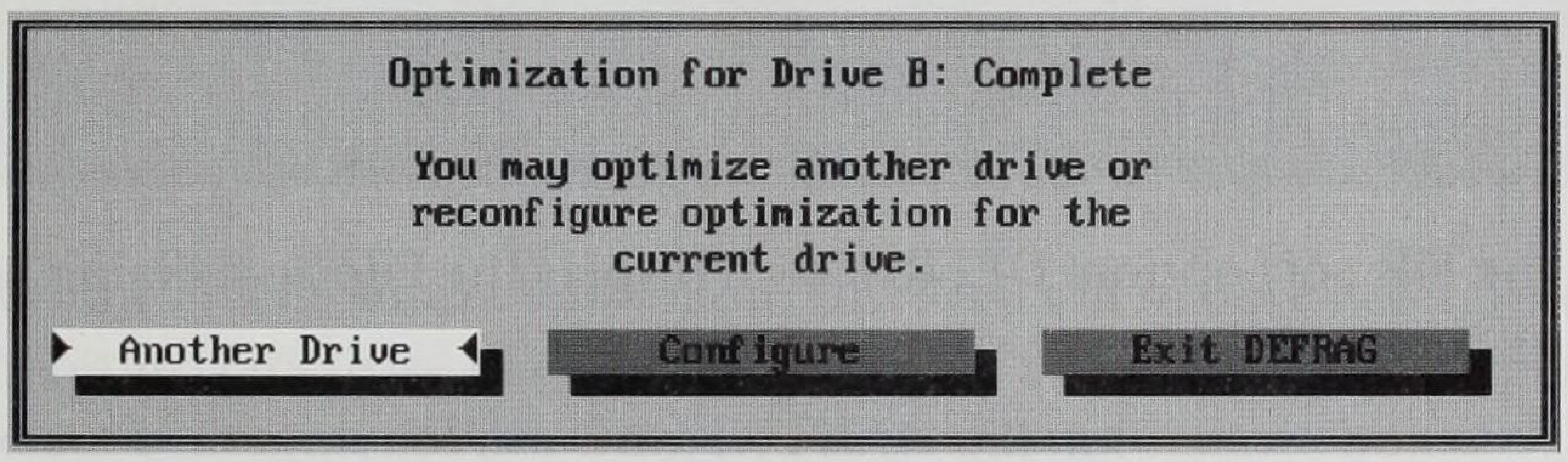
Using your keyboard arrow keys, highlight the desired drive and press ENTER. DEFRAG will analyze the files on your disk, eventually displaying its recommendation, as shown in the Recommendation dialog box:



90% of drive B: is not fragmented.

Recommended optimization method:
Full Optimization

Select the Optimize option. DEFRAG will begin analyzing your files, removing fragmentation. When DEFRAG completes its optimization, it will display a message box. Select OK. DEFRAG will then display the following dialog box asking you if you want to defragment another disk, change DEFRAG's configuration information, or exit. Select the Exit option to return to DOS.

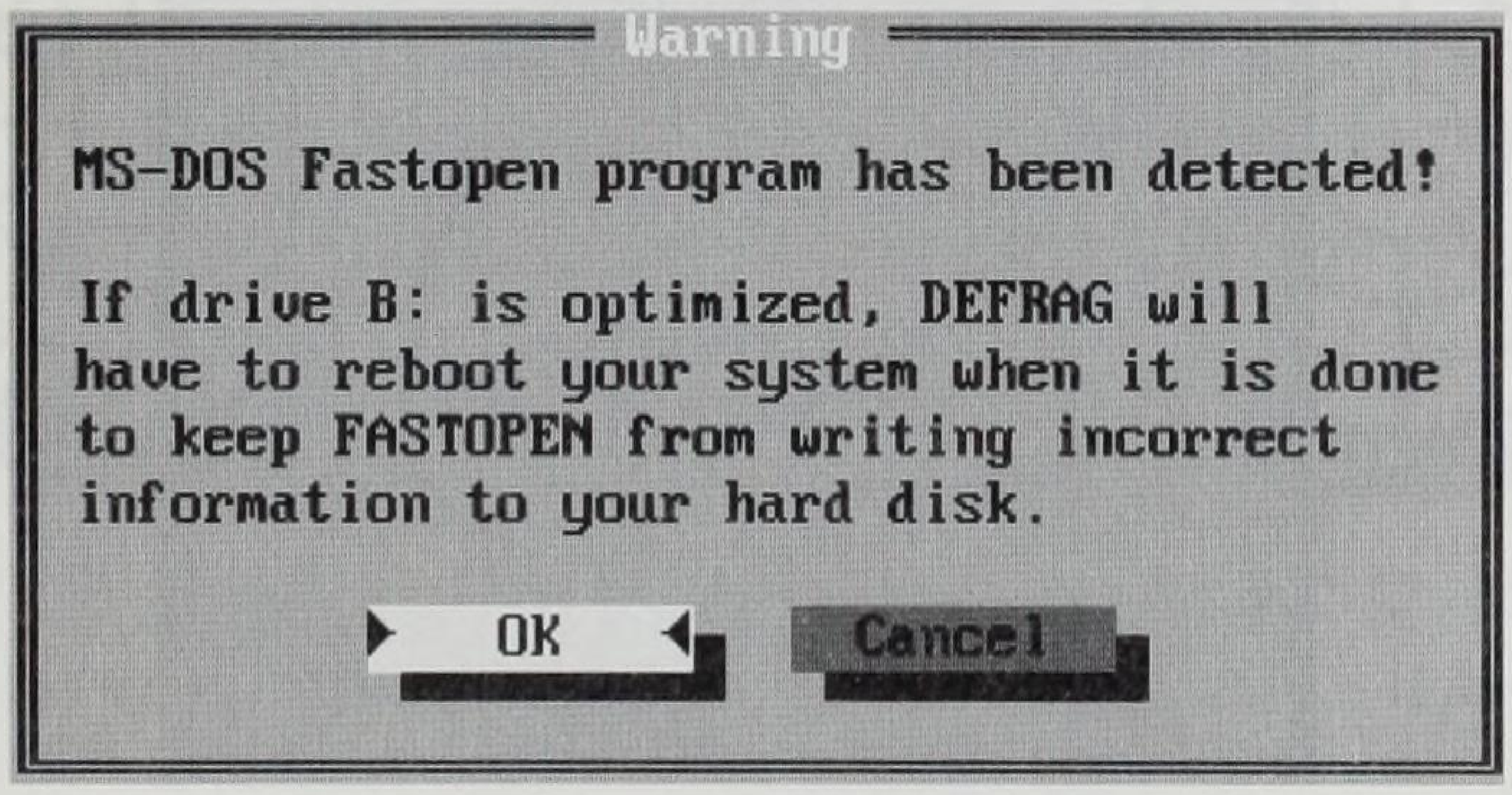


Optimization for Drive B: Complete

You may optimize another drive or
reconfigure optimization for the
current drive.

DEFRAG and FASTOPEN

In Chapter 11 you learned that the DOS FASTOPEN command improves your system performance by keeping information about a file's starting location in RAM. When you use DEFRAG to defragment files, DEFRAG moves files around on the disk. As a result, FASTOPEN's table of file locations is invalid. Unfortunately, the only way to discard FASTOPEN's entries is to reboot your system. As such, if you are using FASTOPEN, DEFRAG will display the Warning dialog box, shown here, when it completes, telling you it will restart your system. Press ENTER to reboot.

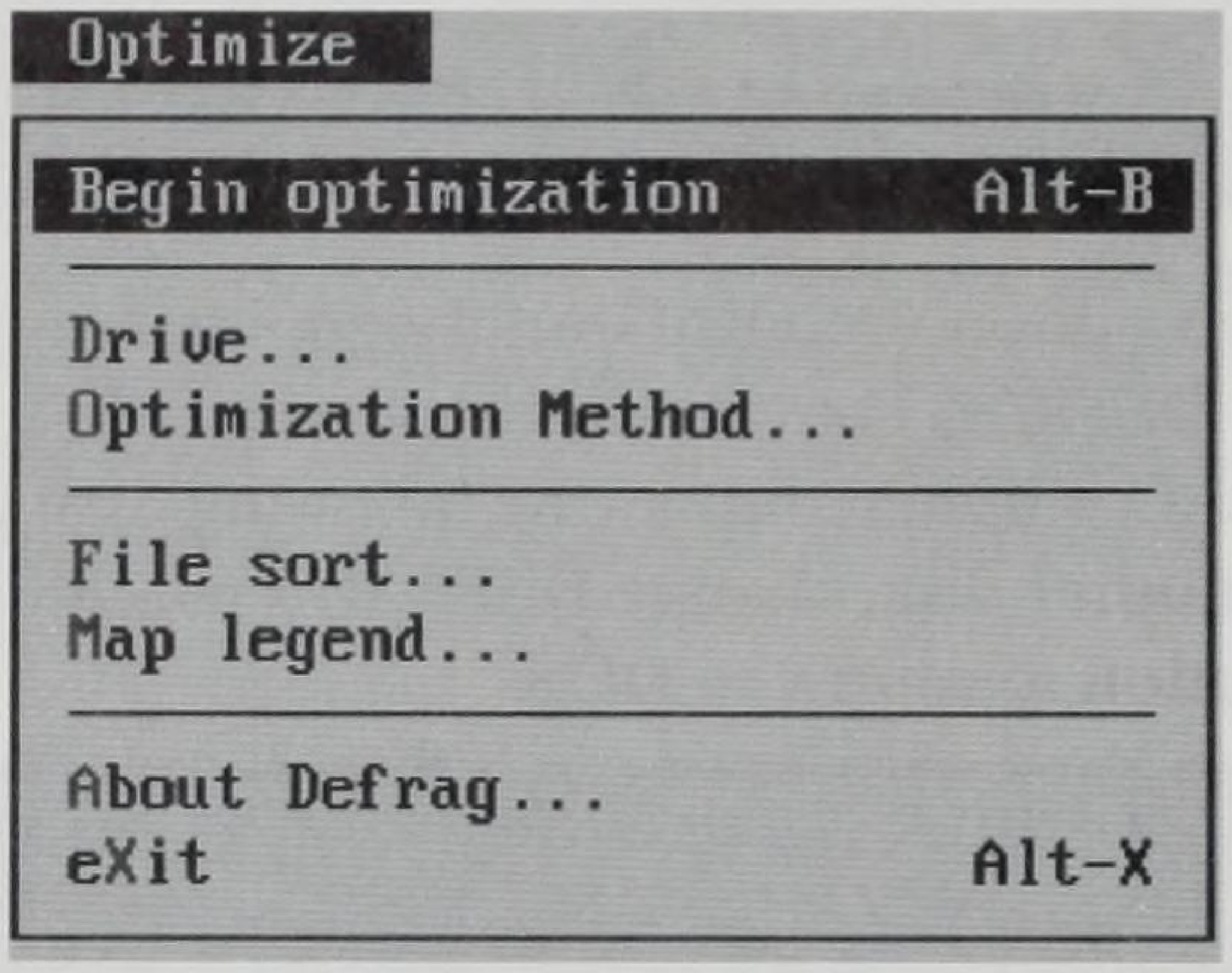


MS-DOS Fastopen program has been detected!

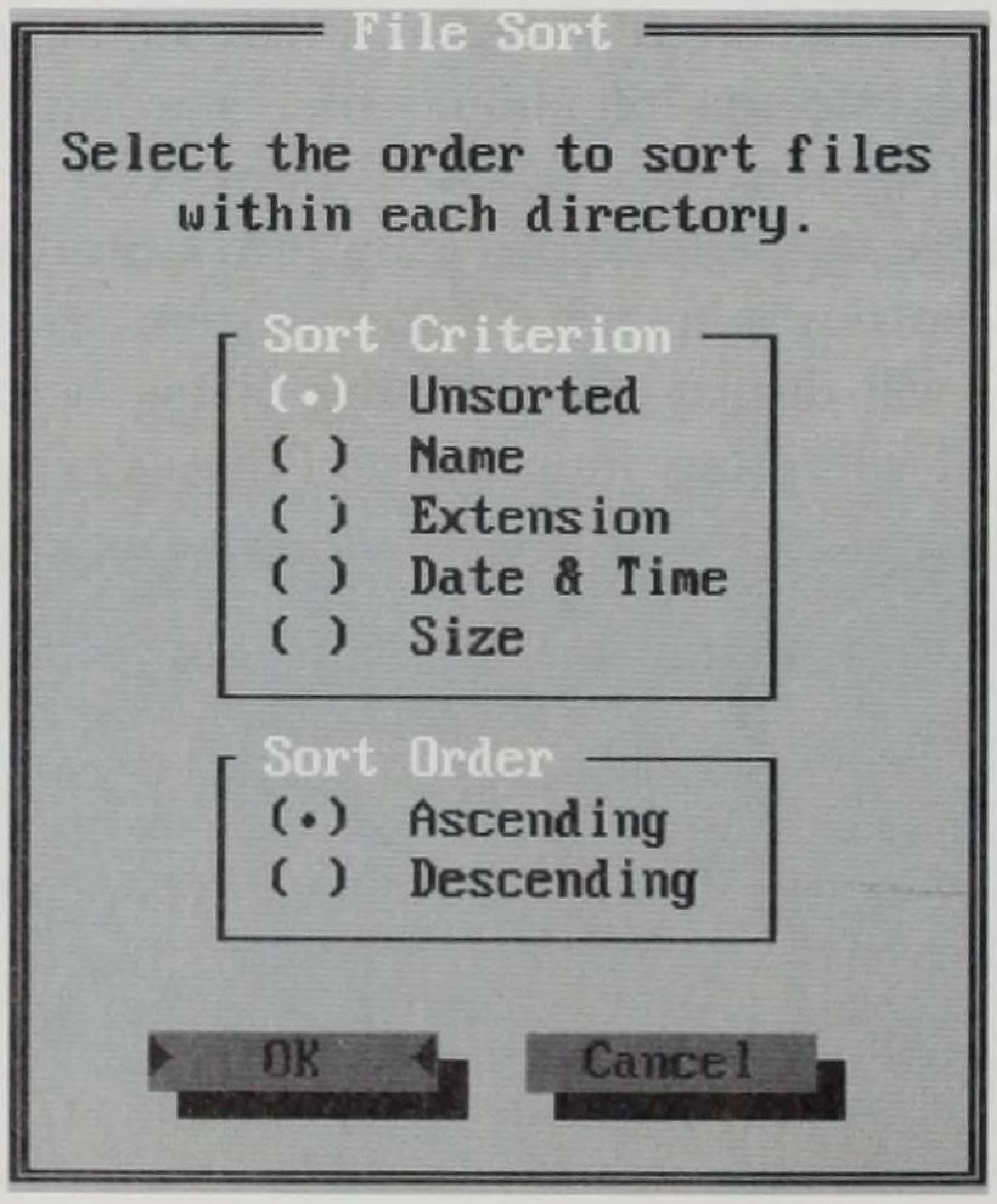
If drive B: is optimized, DEFRAG will
have to reboot your system when it is done
to keep FASTOPEN from writing incorrect
information to your hard disk.

Other DEFRAG Operations

In addition to correcting file fragmentation, DEFRAG also lets you sort the directories on your disk, or remove unused space between files. If you specify a directory sort order, DEFRAG will sort every directory on your disk in the order specified. To specify a sort order, start DEFRAG and select a disk drive as previously discussed. When DEFRAG displays its Recommendation dialog box, use your arrow keys to highlight the Configure option and press ENTER. DEFRAG will display its Optimize menu, shown here:


Optimize
Begin optimization Alt-B
Drive...
Optimization Method...
File sort...
Map legend...
About Defrag...
eXit Alt-X

Select the File sort option. DEFRAG will display the File Sort dialog box, as shown in Figure 42-5.



File Sort

Select the order to sort files within each directory.

Sort Criterion
- (•) Unsorted
- ( ) Name
- ( ) Extension
- ( ) Date & Time
- ( ) Size

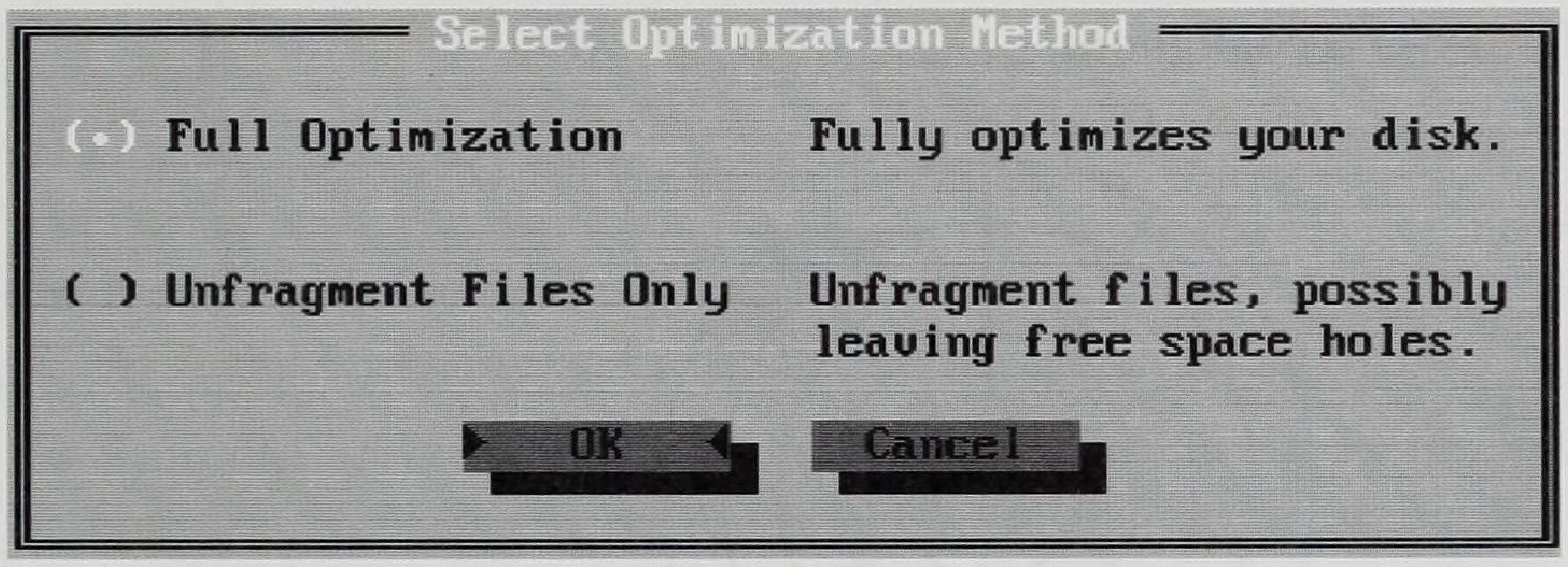
Sort Order
- (•) Ascending
- ( ) Descending

Figure 42-5. *DEFRAG's File Sort dialog box*

Using your arrow keys, highlight the desired sort method. Press ENTER to select OK. DEFRAG will redisplay the Optimize menu. Select the Begin Optimization option. DEFRAG will defragment your files, while sorting your directories as previously discussed.

In addition to defragmenting files, you can direct DEFRAG to pack files together on your disk so that all unused disk space is placed together, as shown in Figure 42-6.

DEFRAG refers to the process of moving unused disk space as removing holes from between files. Using DEFRAG's Optimize menu, you can direct DEFRAG to remove (or leave) holes between files by selecting the Optimization Method option. When you select this option, DEFRAG will display the following dialog box:

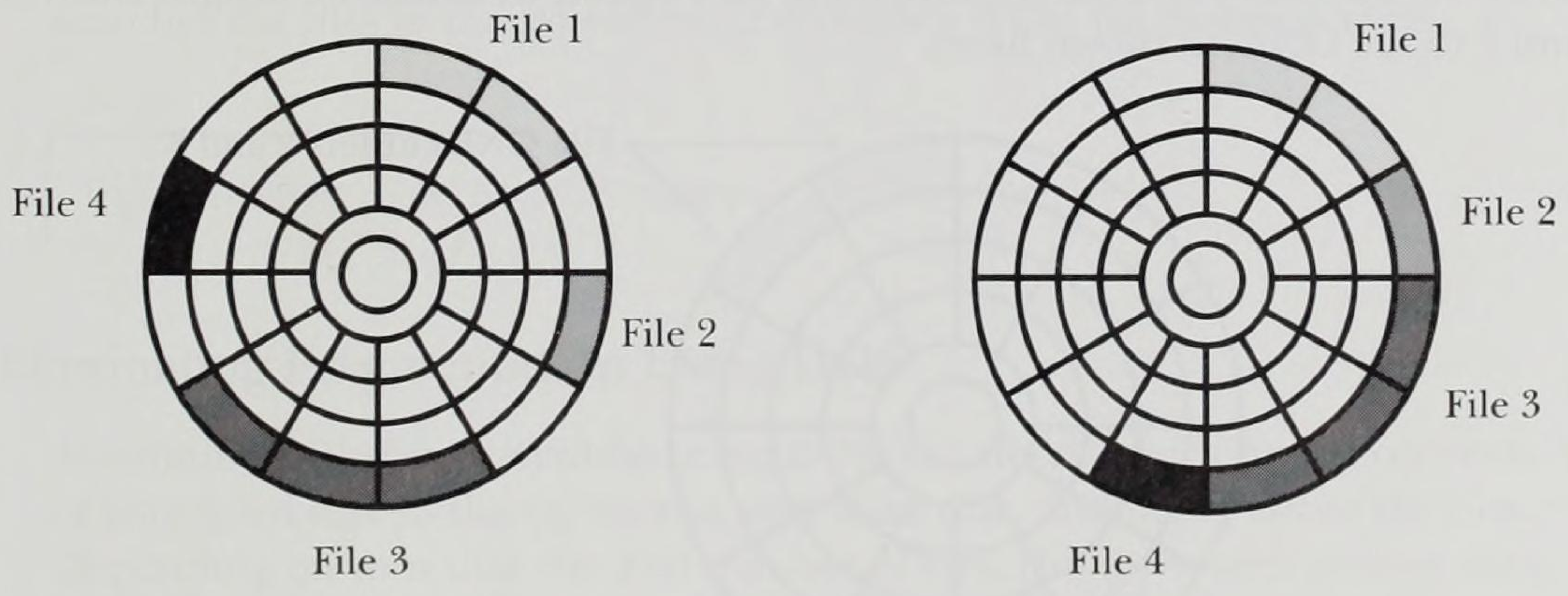


Select Optimization Method

(•) Full Optimization — Fully optimizes your disk.

( ) Unfragment Files Only — Unfragment files, possibly leaving free space holes.

If you select the Full Optimization option, DEFRAG will remove holes from between files. This option may increase the amount of time it takes DEFRAG to optimize your disk. If you select Unfragment Files Only, DEFRAG will not remove holes.



Holes between files

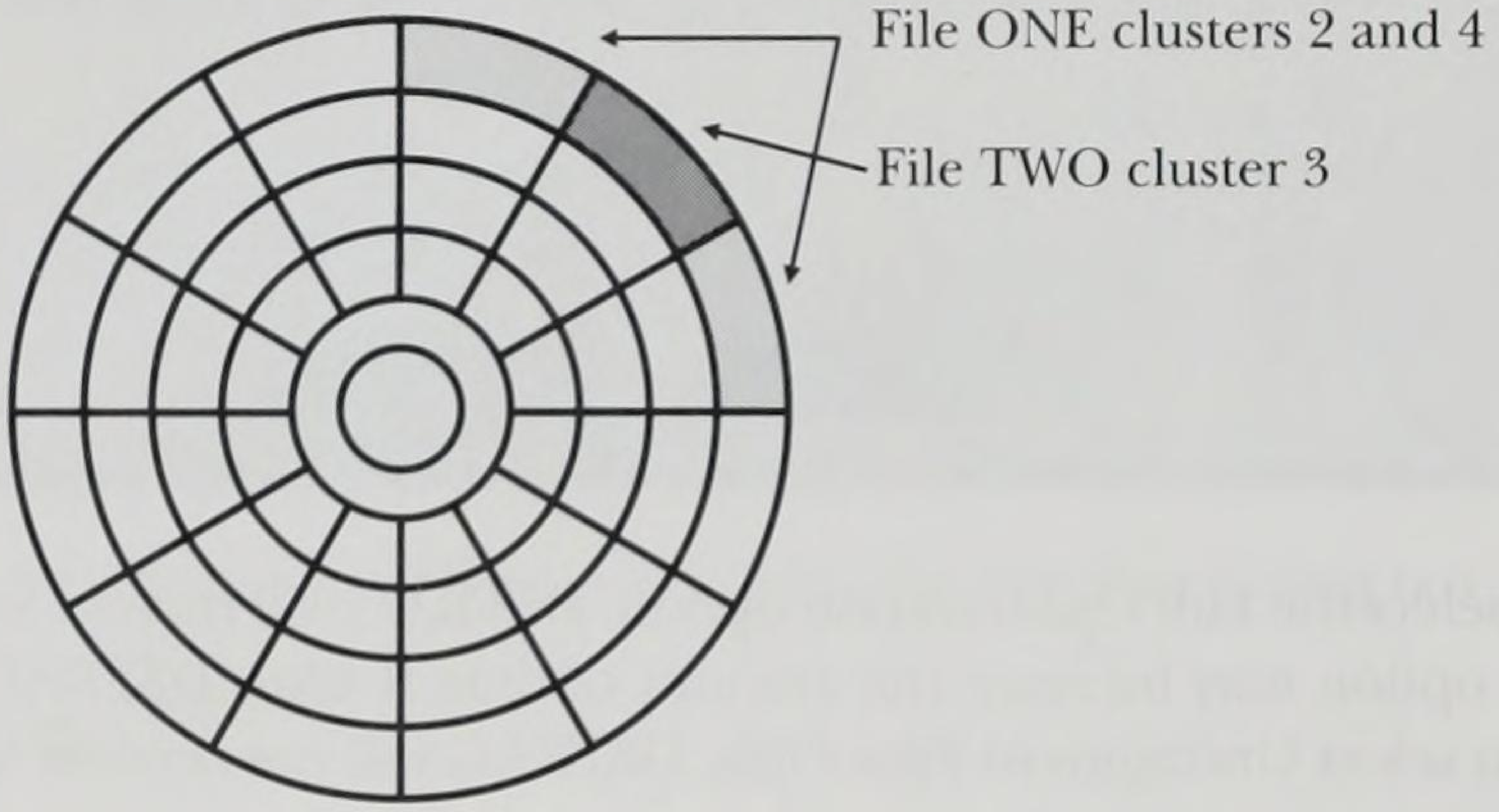
No holes

Figure 42-6. Combining unused disk space

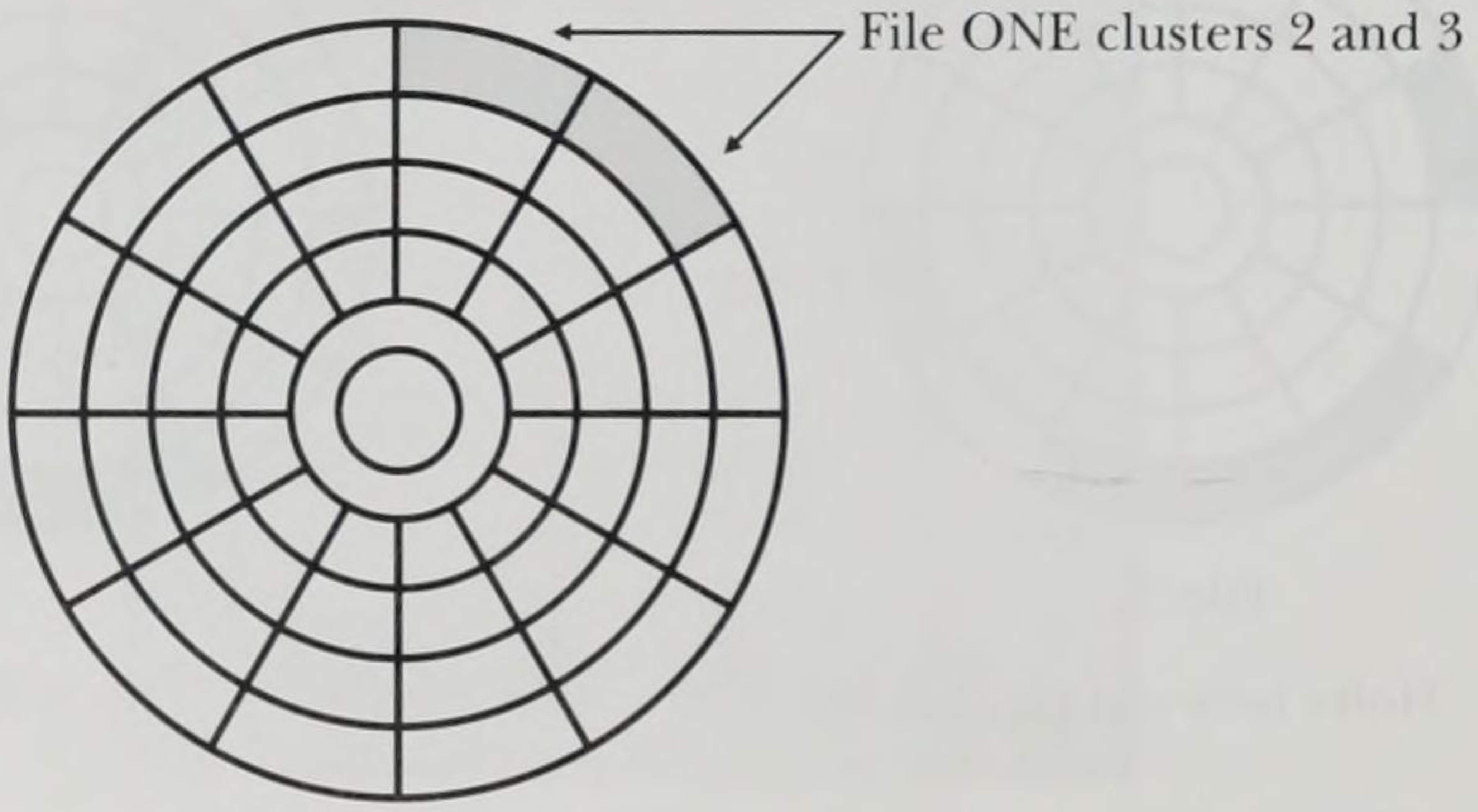
Fragmented Files Can Prevent Successful File Undeletion

If you are using DOS 5 or later, the UNDELETE command lets you recover accidentally deleted files. If you are not using DOS 6, you should upgrade to DOS 6 or purchase a third-party disk utility that provides file undelete capabilities. Your best disk and file protection, however, is current file backups, as discussed in Chapter 22.

If you are using a third-party undelete utility or the DOS 5 or later UNDELETE command, fragmented files may prevent successful file undeletion. Assume, for example, your disk contains the files shown here. In this case, file ONE is fragmented, while file TWO is contiguous.



Assume you accidentally delete both files and you use UNDELETE or a third-party utility program to recover the files. To begin, UNDELETE determines from file ONE's directory entry that file ONE requires two disk clusters. Also, from file ONE's directory entry, UNDELETE knows the file's starting cluster. Unfortunately, UNDELETE assumes file ONE is contiguous. As a result, UNDELETE assigns clusters 2 and 3 to file ONE, as shown here:



Because UNDELETE has incorrectly assigned cluster 3 to file ONE, you cannot use UNDELETE to recover file TWO since UNDELETE knows another file has overwritten file TWO's contents. (File TWO's starting cluster is now in use by file ONE.) The net result of the UNDELETE operation is that file ONE contains the wrong information, file TWO cannot be recovered, and file ONE's actual second cluster cannot be recovered.

If you use the MIRROR command's delete tracking, discussed in Chapter 28, UNDELETE may correctly recover fragmented files. For most undelete utilities, however, fragmented files pose a great difficulty.

For best performance and the best chance of undeleting files, check your disk for fragmented files and eliminate the fragmentation on a regular basis.

SUMMARY

Fragmented Files

A fragmented file is a file whose contents are stored in clusters spread out across your disk. Fragmented files decrease your system performance by increasing disk read/write head movements and disk rotation delays. File fragmentation is a natural occurrence of creating, deleting, and enlarging files on your disk. If your programs start slower, or your data files take longer to load, your files may be fragmented.

Using CHKDSK to Detect Fragmentation

If you include a filename or DOS wildcards in the CHKDSK command line, CHKDSK will examine the matching files for fragmentation. If a file is fragmented, CHKDSK will display a message informing you of the number of noncontiguous clusters the file contains. The following CHKDSK command searches the files in the WordPerfect directory WP51 for fragmentation.

```
C:\> CHKDSK \WP51\*.* <ENTER>
```

Eliminating Fragmentation Using DOS

To eliminate file fragmentation using DOS, you need to back up the contents of your hard disk to floppy, format your hard disk, and then restore the files. Depending on your disk size and number of files, this three-step process may become quite time consuming. (To simplify defragmentation, you can use one of many third-party utility programs.) If you are using DOS 6, you can quickly eliminate fragmented files using the DEFRAG command.

Hands On

CHKDSK lets you examine a directory or file for fragmentation. The following CHKDSK command checks the files in the DOS directory:

```
C:\> CHKDSK  \DOS\*.* <ENTER>
```

Using CHKDSK in this way, DOS examines the files in your most commonly used directories. If CHKDSK lists two or three files, don't worry. If CHKDSK lists a large number of files, it's probably a good time to buy and use a third-party defragment utility or to perform a BACKUP, FORMAT, and RESTORE operation.

Review

1. What is a fragmented file? What are its symptoms?
2. How does fragmentation occur?
3. How can you identify fragmented files?
4. How can you correct fragmentation?

Answers

1. A fragmented file is a file whose contents are stored in sectors dispersed across your disk. When a program starts slowly, or a file (such as a word processing document or spreadsheet) takes a long time to load, the file may be fragmented.
2. Fragmentation is a natural result of creating, deleting, and increasing the size of files on your disk. You cannot prevent file fragmentation, you can only correct fragmentation once it occurs.
3. The CHKDSK command lets you examine the files in a directory for fragmentation. The following command, for example, checks the files in the root directory:

```
C:\> CHKDSK  \*.* <ENTER>
```

4. Using DOS versions earlier than 6, you can correct fragmentation by performing a complete disk backup, format, and restore operation. If you are using DOS 6, you can quickly eliminate fragmented files by using the DEFRAG command. Another method for correcting fragmentation is to use a third-party disk-utility program.

Advanced Concepts

To fully understand how file fragmentation affects your system performance, you need to break a disk read or write operation into individual parts and examine the time required for each.

Disk manufacturers often advertise their disk's average *access time,* in other words, the time to read or write a disk sector. Disk access time is made up of four parts: seek time, rotational delay, data-transfer time, and controller overhead.

Seek time is the amount of time the controller takes to move the disk's read/write head to the desired track. Most disk manufacturers will advertise an average seek time based on benchmark programs, typically around 20ms (milliseconds).

Rotational delay is the amount of time the disk controller must wait for the desired sector to spin past. All hard disks will have this rotational delay. Hard disks all spin at 3600 RPM (rotations per minute). To determine the average rotational delay, simply compute the amount of time required to spin the disk one half revolution, as illustrated in Figure 42-7.

Data-transfer time is the amount of time required to transfer a sector of data to or from disk. Transfer rates of 2 to 4Mb per second are not uncommon.

Last, to read or write data, the disk controller must perform its own processing. This *controller overhead* typically takes only a few milliseconds.

Because the controller overhead and data transfer times are fixed and the disk rotation speed is always 3600 RPM, the only way to reduce the average disk access

$$\frac{\text{3600 revolutions}}{\text{minute}} \div \frac{\text{1 minute}}{\text{60 seconds}} = \frac{\text{60 revolutions}}{\text{second}}$$

1 revolution = 0.0166 seconds

1/2 revolution = 0.00833 seconds or 8.3 ms

Figure 42-7. Computing a hard disk's average rotational delay

time is to reduce the number of seeks and rotations the disk drive must perform. By eliminating file fragmentation, you do just that.

Key Terms

Fragmented file A file whose contents are stored in two or more disk clusters spread out across your disk.

Contiguous file A file whose contents are stored in consecutive disk clusters.

Disk head seek The process of moving the disk read/write head from one disk track to another.

Rotational delay The amount of time the disk controller must wait for the desired sector to spin past the disk drive's read/write head.

Seek time The amount of time required to move the disk head from the current track to the desired track.

Chapter 43

Disk Partitions and FDISK

There are four steps involved in preparing a hard disk for use by DOS. The first step is telling your computer about your disk, its size, and its characteristics. If you are using an IBM PC AT or AT-compatible, this step involves running the SETUP program that accompanied your computer on floppy disk. If you are using a 386 or 486 computer, the SETUP program is typically built into the computer's read-only memory (ROM). In either case, the documentation that accompanied your computer should explain the SETUP process. Chapter 35 briefly discussed the second step, low-level formatting. In most cases, the computer manufacturer will perform this for you. The third step is dividing your disks into one or more partitions using FDISK. This chapter examines the FDISK command and disk partitions in detail. As before, depending on your computer manufacturer, your disk may have been partitioned for you. The final step is formatting your disk using the FORMAT command, discussed in Chapter 7. If your computer was able to start DOS the first time you turned it on, all four steps have been performed for you.

Understanding Hard Disk Partitions

When the original IBM PC shipped in 1981, it did not support hard disks. When IBM released the PC XT, which supported hard disks, the disks were typically only 10Mb. For several years after the first hard disk was placed in a PC, the cost of hard disks kept their size in the 10- to 20-megabyte range. A user with a large budget could afford a 30Mb disk. Because hard disks were typically 30Mb or less, DOS only supported sizes up to 32Mb.

Eventually, disk technology caught up with the rest of the PC industry, and disks larger than 32Mb became readily affordable. Unfortunately, the largest disk size DOS could support was 32Mb. To let users with disks larger than 32Mb use the entire disk, DOS 3.3 let the user divide the disk into sections, called *partitions.* The first partition became drive C; the second partition, drive D. FDISK lets you create from one to four partitions on your disk. DOS, however, will only use two partitions, one called the *primary partition,* from which you start DOS, and a second called the *extended partition.* If you are using a second operating system in addition to DOS, you might create a third or fourth partition to store the operating system's files. If you are only using DOS, you will only create two partitions at most, a primary DOS partition and an extended partition.

As you know, a hard disk contains hundreds of circular tracks (or cylinders). A partition is simply a collection of consecutive tracks, as illustrated in Figure 43-1.

Conceptually, a partition is a barrier DOS cannot cross, much like the partitions that separate traffic on a highway. A command that works on one partition, such as FORMAT or DEL, cannot affect information stored on a second partition.

Prior to DOS 4, disk partitions where restricted to 32Mb or less. DOS 4 increased the maximum partition size to 512Mb. In DOS 5, partition sizes grew to a maximum size of 2Gb (gigabytes)!

The FDISK command partitions your disk. If you are only using DOS, you have two choices. First, provided your DOS version supports large partitions, you can use your entire disk as a single partition. This is the easiest technique, also the most

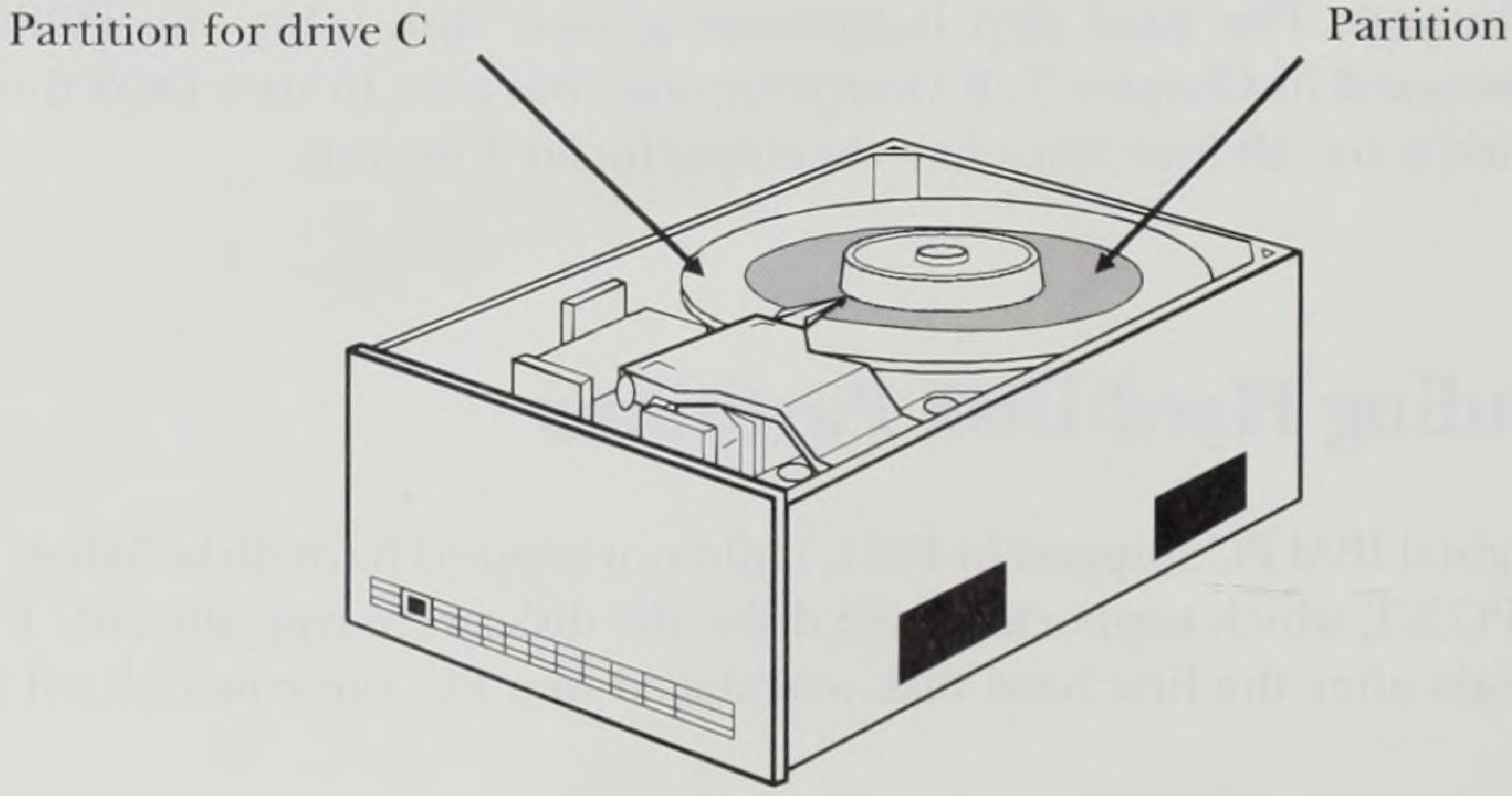


Figure 43-1. *Partitions divide your disk into distinct regions*

common. Second, you can use FDISK to divide your disk into two partitions, a primary DOS partition and an extended partition. Your primary DOS partition will become drive C. Your extended DOS partition will contain the rest of your disk (possibly more than 32Mb). Once you define your extended partition, you must further divide the partition into logical drives, each smaller than or equal to the largest partition size DOS supports. Figure 43-2 illustrates a hard disk with a primary and extended partition, for which the extended partition has been further divided into drives D, E, and F.

Other Reasons to Partition a Hard Disk

You can divide your disk in up to four partitions. However, DOS only lets you use two partitions, a primary partition and an extended partition. The remaining two partitions can be used for another operating system.

DOS is only one of several operating systems that run on the IBM PC or PC-compatibles. Some users, therefore, create a partition for DOS and a partition for a second operating system such as OS/2, UNIX, or XENIX (a version of UNIX developed by Microsoft). By separating the operating systems into different partitions, the user is free to boot and use any operating system without affecting another operating system's files in a different partition.

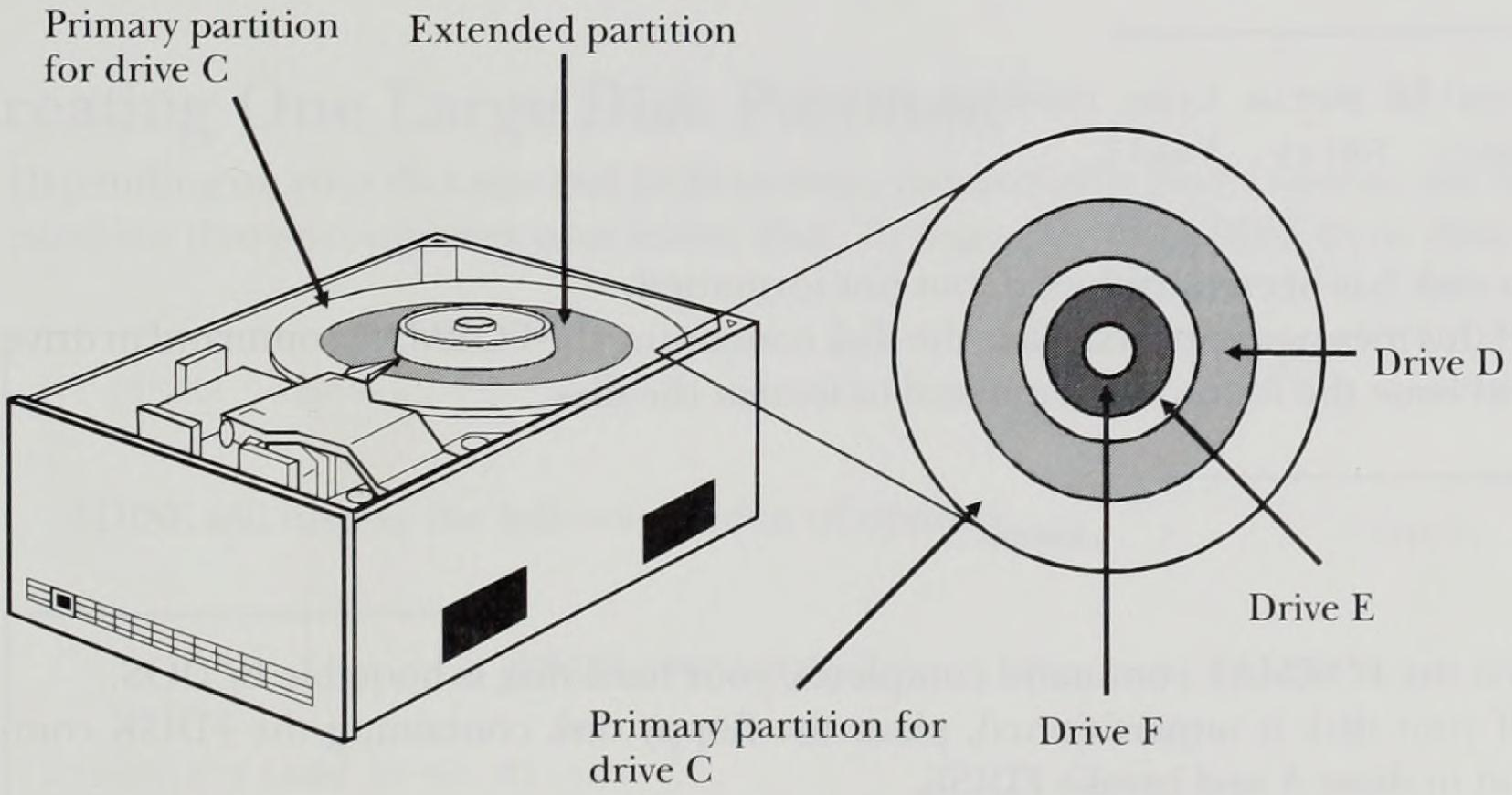


Figure 43-2. *A hard disk with a primary and an extended partition, where the extended partition is further divided into the logical drives D, E, and F*

Using FDISK

If you can start your computer from your hard drive, your disk has been partitioned and formatted. If you cannot start your computer from hard drive, you may need to tell your computer about the hard drive using SETUP, and then partition and format the drive. In this case, you will need to first boot DOS using a bootable floppy disk.

Your hard disk may have already been partitioned for you. To determine if your disk has been partitioned, issue the following command.

```
A> DIR C: <ENTER>
```

If DOS displays the following message, your disk has not yet been partitioned.

```
Invalid drive specification
```

If, instead, DOS displays one of the following error messages, your disk has been partitioned, but not formatted.

```
General failure reading drive C
Abort, Retry, Fail?
```

or

```
Invalid media type reading drive C
Abort, Retry, Fail?
```

Your disk has been partitioned, but not formatted.

If this message appears, place the disk containing the FORMAT command in drive A and issue the following command to format the disk.

```
A> FORMAT C: /S <ENTER>
```

When the FORMAT command completes, your hard disk is bootable by DOS.

If your disk is unpartitioned, place the floppy disk containing the FDISK command in drive A and invoke FDISK.

```
A> FDISK <ENTER>
```

If FDISK displays the following message, your computer does not know about your hard disk.

```
No fixed disk present
```

If this message (or one similar) occurs, you must invoke your computer's setup program to inform the computer about the disk. For IBM AT computers, the setup program typically resides on a floppy disk that accompanied your computer. For 386 and 486 systems, the setup is typically built into the computer's ROM. To access the built-in setup, you press a specific key, such as DEL, during the computer's power-on self-test. Whether you are using a setup program on disk or one built into your computer, the documentation that accompanied your computer will explain the steps you must follow to complete the setup operation.

The setup program accesses and modifies a special memory area of your computer called *CMOS.* The CMOS is a battery-powered chip that stores information about your computer, such as the number and type of disks.

Depending on whether you are using a large single partition or a primary and extended partition, the steps you must follow to partition your disk will differ.

NOTE

Never use FDISK on a disk containing data. To partition a disk, FDISK overwrites portions of the disk, possibly destroying information you need. In DOS 5 or later, if you inadvertently overwrite partition table information, you may be able to recover your disk provided you have used the MIRROR /PARTN command, discussed in Chapter 28.

Creating One Large Disk Partition

Depending on your disk size and DOS version, you may only have to create one large partition that encompasses your entire disk. To begin, invoke FDISK from drive A.

```
A> FDISK <ENTER>
```

FDISK will display the following menu of options.

```
                        FDISK Options

Current fixed disk drive: 1

Choose one of the following:
```

```
1. Create DOS Partition or Logical DOS Drive
2. Set active partition
3. Delete DOS Partition or Logical DOS Drive
4. Display partition information

Enter choice: [1]


Press ESC to exit FDISK
```

To create the DOS partition, type 1 and press ENTER. Note that FDISK displays the default menu response between the left and right brackets. When you press ENTER to select option 1, FDISK displays the following menu.

```
          Create DOS Partition or Logical DOS Drive

Current fixed disk drive: 1

Choose one of the following:

1. Create Primary DOS Partition
2. Create Extended DOS Partition
3. Create Logical DOS Drive(s) in the Extended DOS Partition

Enter Choice: [1]

Press Esc to return to FDISK Options
```

To create the partition, select option 1 and press ENTER. FDISK will display the following screen, asking you to specify the partition's desired size.

```
          Create Primary DOS Partition

Current fixed disk drive: 1

Do you wish to use the maximum available size for a Primary DOS
partition and make the partition active (Y/N).............[Y]


Press Esc to return to FDISK Options
```

In this case, you want FDISK to use your entire disk for a partition. Type Y and press ENTER. FDISK will create the partition on disk, displaying the following message to tell you it will restart the system.

```
System will now restart

Insert DOS diskette in drive A
Press any key when ready...
```

Place your bootable DOS floppy disk in drive A and press any key to continue. When your system restarts, use the FORMAT command to prepare drive C as a bootable disk as follows:

```
A> FORMAT  C:  /S  <ENTER>
```

Creating a Primary and an Extended Partition

If your DOS version does not support partitions as large as your disk size, you must create a primary and an extended partition. Before you begin, you need to plan out how you want your disk partitioned. As briefly discussed, the extended disk partition can be any size. Within the extended partition you must divide the disk space into one or more logical drives, whose maximum size is less than the largest partition size DOS supports.

When you use a primary and an extended partition, your primary partition becomes drive C. Your extended partition can take on the drive letters D through Z (up to 23 logical drives!), depending on your partitioning.

If you make your partition or logical disk drive sizes too small, your drives will run out of disk space quickly. In addition, you may constrain the size of the largest file you can create.

As before, invoke FDISK from the floppy drive in disk A. FDISK will display its opening menu.

```
                   FDISK Options

Current fixed disk drive: 1

Choose one of the following:

1. Create DOS Partition or Logical DOS Drive
```

```
2. Set active partition
3. Delete DOS Partition or Logical DOS Drive
4. Display partition information

Enter choice: [1]

Press ESC to exit FDISK
```

Select option 1 to create a DOS partition. FDISK will display the following menu to determine if you want to create the primary or extended partition.

```
          Create DOS Partition or Logical DOS Drive

Current fixed disk drive: 1

Choose one of the following:

1. Create Primary DOS Partition
2. Create Extended DOS Partition
3. Create Logical DOS Drive(s) in the Extended DOS Partition

Enter Choice: [1]

Press Esc to return to FDISK Options
```

FDISK will not let you create an extended partition until after you create the primary partition. Select option 1 to create a primary partition. FDISK will display the following message, asking you if you want to use the largest possible partition.

```
          Create Primary DOS Partition

Current fixed disk drive: 1

Do you wish to use the maximum available size for a Primary DOS
partition and make the partition active (Y/N).............[Y]


Press Esc to return to FDISK Options
```

If you type **Y**, FDISK will create as big a primary partition as your disk will hold and your DOS version will support, display the following message, and restart when you press a key.

```
System will now restart

Insert DOS diskette in drive A
Press any key when ready...
```

After your system restarts, use the following command to prepare the primary partition for use by DOS.

```
A> FORMAT  C: /S  <ENTER>
```

Now you must create your extended DOS partition and define the corresponding logical drives. As before, invoke FDISK and select option 1 to create a DOS partition. When FDISK displays the following menu, select option 2 to create an extended partition.

```
          Create DOS Partition or Logical DOS Drive

Current fixed disk drive: 1

Choose one of the following:

1. Create Primary DOS Partition
2. Create Extended DOS Partition
3. Create Logical DOS Drive(s) in the Extended DOS Partition

Enter Choice: [1]

Press Esc to return to FDISK Options
```

FDISK will display a screen similar to the following that prompts you for the extended partition size.

```
          Create Extended DOS Partition

Current fixed disk drive: 1

Partition Status Type   Size in Mbytes   Percentage of Disk Used
 C:1        A    Pri DOS       30             50%

Total disk space is 60 Mbytes (1 Mbyte = 1048576 bytes)
```

```
Maximum space available for partition is 30Mbytes (50%)

Enter partition size in Mbytes or percent of disk space (%) to
create an Extended DOS Partition..........................[ 30]


Press Esc to return to FDISK options
```

Press ENTER to select the default partition size, which consumes the remainder of the disk. FDISK will display the following screen, telling you it has successfully created the partition.

```
          Create Extended DOS Partition

Current fixed disk drive: 1

Partition Status Type   Size in Mbytes   Percentage of Disk Used
 C:1        A    Pri DOS       30             50%
   2             Ext DOS       30             50%


Extended DOS Partition created

Press Esc to continue
```

You must further divide your extended partition into logical drives. Press ESC to continue, and FDISK will display a message telling you that no logical drives exist and prompting you to type in the size of the first drive.

```
Create Logical DOS Drive(s) in Extended DOS Partition

No logical drives defined

Total Extended DOS Partition is 30 Mbytes (1 Mbyte = 1048576
   bytes)
Maximum space available for logical drive is 30Mbytes (100%)


Enter logical drive in Mbytes or percent of disk space (%)..[30]

Press Esc to return to FDISK options
```

When you type in the size and press ENTER, FDISK will create the logical drive. Depending on the size of your extended partition, FDISK may prompt you for several logical drives. After you have defined all the logical drives, press ESC to return to FDISK's main menu and then ESC to return to DOS. As before, FDISK will display a message telling you the system will restart. Press any key to restart the system.

After the system successfully restarts, use the following format command to prepare the logical disk D for use by DOS.

```
C:\> FORMAT  D:  <ENTER>
```

Note the FORMAT command does not include /S. Because DOS only boots off the primary DOS partition, formatting the extended partitions as bootable disks would consume space unnecessarily for the DOS system files.

Hands On

If you don't need to partition your disk, there is no need for you to invoke FDISK. If you have worked with DOS for some time and have a current backup copy of your disk, you can invoke FDISK to determine the cylinders used for each of your disk partitions.

```
C:\> FDISK  <ENTER>
```

At the FDISK main menu, select option 4 to display partition information. FDISK will provide information about your primary and extended partitions and then ask you if you want to view information about the logical drives. If you type **Y**, FDISK will display the starting and ending cylinder number for each logical drive along with the drive's size. Press ESC twice to return to DOS.

Review

1. What is a partition?
2. How many partitions can a disk have? How many of these can DOS use?
3. What is a logical drive?
4. What are the four steps to prepare a hard disk for use?

5. What is the maximum primary partition size in the following DOS versions: 3.3, 4, 5?

Answers

1. A partition is a region of consecutive cylinders on your disk logically separated from the information stored on other partitions on the same disk.
2. A disk can contain up to four partitions. DOS lets you use two of these partitions, which are a primary DOS partition and an extended partition.
3. If you create an extended partition on your disk, you must divide the extended partition into one or more logical drives. Your primary DOS partition will be drive C, and your first logical drive will be drive D. Because you only really have one hard drive, these additional (nonphysical) drives are called logical drives.
4. The four steps to prepare a hard disk are
    a. Inform your computer about the disk using SETUP.
    b. Perform a low-level format operation on the disk.
    c. Partition the disk using FDISK.
    d. Perform a high-level or logical disk format using FORMAT.
5. DOS 3.3 supports a maximum partition size of 32Mb, DOS 4 supports a maximum partition size of 512Mb, and DOS 5 supports a maximum partition size of 2Gb.

Advanced Concepts

There may be times when you want to change your disk's current partitions, for example, to upgrade to a version of DOS that supports larger partitions. To begin, back up your primary partition and logical drives to floppies. When the backup operation completes, invoke FDISK.

When FDISK displays its main menu, select option 3 to delete the DOS partitions. FDISK will display the following menu:

```
           Delete DOS Partition

Current fixed disk drive: 1

Choose one of the following:

1. Delete Primary DOS Partition
2. Delete Extended DOS Partition
3. Delete logical drive(s) in the Extended DOS Partition
4. Delete Non-DOS Partition

Enter choice: []
Press Esc to return to FDISK options
```

To begin, you must first delete your logical drives. Select option 3 and press ENTER. FDISK will display a screen containing information about each logical drive, followed by a prompt for the drive letter you want to delete. Type in the drive letter and press ENTER. FDISK will mark the drive as deleted and prompt you for the next drive letter. Repeat this process until no other logical disks exist. Press ESC to return to FDISK's main menu. Select option 3 again. FDISK will display a menu similar to the previous one. Choose option 2 to delete the extended partition. FDISK will display a warning message telling you the information on the partition will be lost and asking you to verify you want to continue. Type **Y** to delete the partition and press ENTER. FDISK will delete the partition. Press ESC to return to FDISK's main menu. Again select option 3. FDISK will display the previous main menu. Select option 1 to delete the primary DOS partition. As before, FDISK will display its warning that data will be lost. Type **Y** and press ENTER to delete the partition. After FDISK deletes the partition, press ESC to return to FDISK's main menu. You can now create a new primary and, if necessary, a new extended partition using the techniques presented earlier in this chapter.

Where FDISK Records Partition Information

As you have learned, every DOS disk has a boot record that it reads to determine the disk size and possibly to begin the DOS startup process. In addition, every hard disk contains a master boot record, which contains the disk's partition information, such as the starting cylinder for each partition. The master boot record starts at the disk's very first cylinder. The DOS boot record, on the other hand, starts at the beginning of the DOS partition, as illustrated in Figure 43-3.

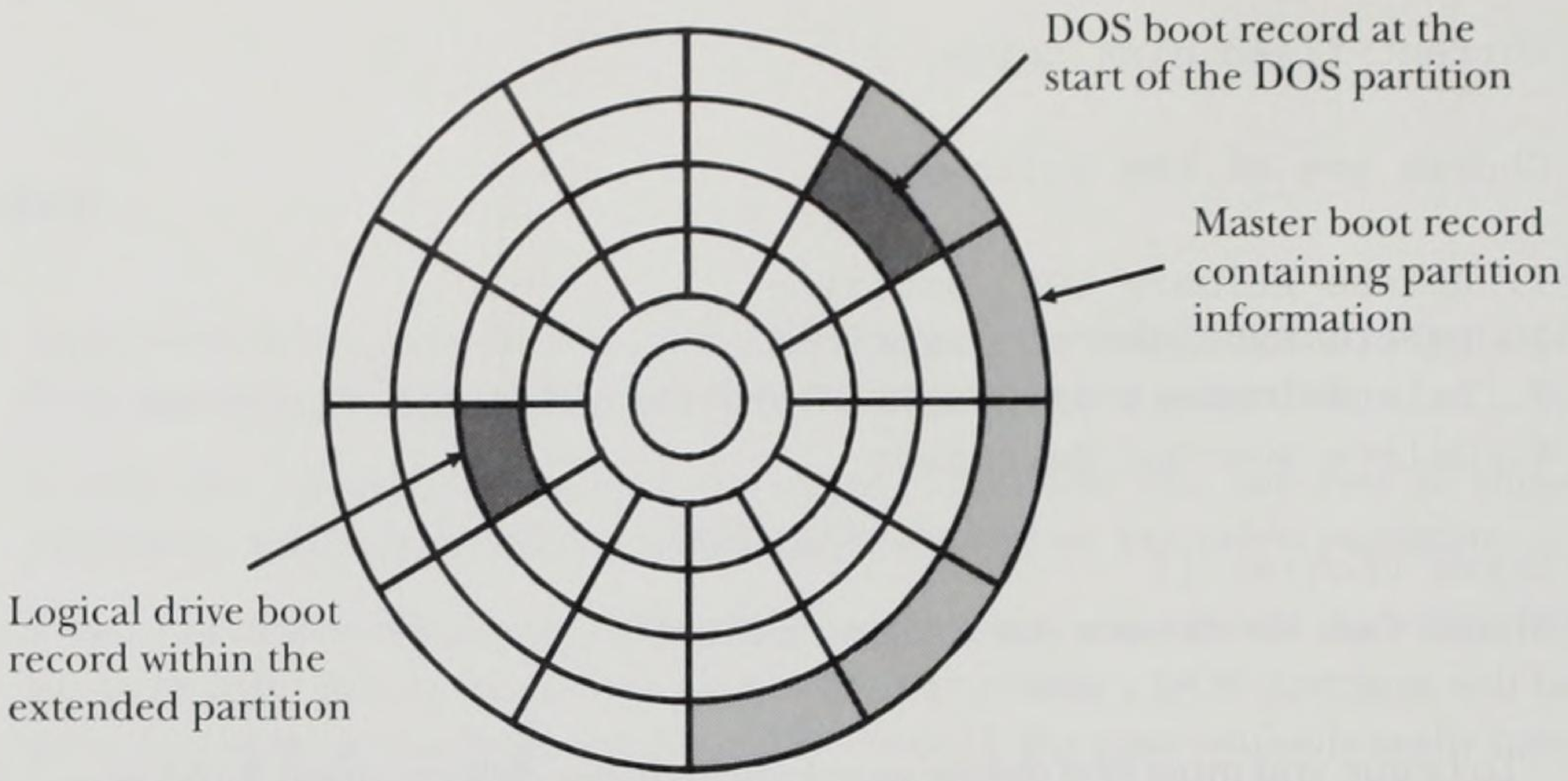

DOS boot record at the start of the DOS partition
Master boot record containing partition information
Logical drive boot record within the extended partition

Figure 43-3. *Every hard disk contains a master boot record that stores the partition information*

When you use FDISK to create or change disk partitions, FDISK updates the partition information stored in the disk's master boot record.

Key Terms

Partition A region of consecutive, logically separated cylinders on a hard disk.

Primary partition The disk partition used to boot DOS.

Extended partition A technique introduced by DOS 3.3 to access disks larger than 32Mb. The extended partition could hold all the disk space unused by the primary partition. The extended partition was then further divided into logical disk drives, whose size could not exceed the maximum partition size defined by DOS.

Logical drives Nonphysical disk drives. Your computer probably only has one hard drive. If you create an extended partition on the drive, you must further divide the partition into logical drives. Each of these drives receives a drive letter, such as D, E, F, all the way up to Z. Because your disk really doesn't have this many drives, these drives are called logical drives.

Chapter 44

Doubling Your Disk's Storage Capacity

If you are using DOS 6, the DBLSPACE command lets you compress the information stored on your hard disk into a more compact format. In this way, DBLSPACE essentially doubles your disk's storage capacity. If you are using a 100Mb hard disk, for example, running DBLSPACE might increase the disk's storage capacity to 200Mb! Best of all, the compression is transparent to your work with DOS. You create, edit, rename, and delete files using the same commands you've always used. In general, the DOS 6 DBLSPACE command doubles the storage capacity of your hard disk without impacting your work.

Doubling Your Hard Disk's Storage Capacity

One of the first things you will probably want to do after you install DOS 6 is to invoke DBLSPACE to double your hard disk's storage capacity. Although the Microsoft documentation does not direct you to do so, you should probably first back up your disk's existing contents, as discussed in Chapter 22. After the backup operation successfully completes, invoke DBLSPACE from the DOS prompt as in the following:

```
C:\> DBLSPACE <ENTER>
```

DBLSPACE will display its initial screen as shown here:

```
Microsoft DoubleSpace Setup
===========================

       Welcome to DoubleSpace Setup.

       The Setup program for DoubleSpace frees space on your hard
       disk by compressing the existing files on the disk.
       Setup also installs the DoubleSpace device driver to manage
       the compressed data.

         o To set up DoubleSpace now, press ENTER.

         o To learn more about DoubleSpace Setup, press F1.

         o To quit Setup without installing DoubleSpace, press F3.
```

Press ENTER to continue. DBLSPACE will display the following menu of options:

```
Microsoft DoubleSpace Setup
===========================

      There are two ways to run Setup:

      Use Express Setup if you want DoubleSpace Setup to compress
      drive C and determine the compression settings for you.
      This is the easiest way to install DoubleSpace.

      Use Custom Setup if you are an experienced user who wants to
      specify the compression settings and drive configuration
      yourself.

      ┌──────────────────────────────────────────────────────────────┐
      │      Express Setup (recommended)                             │
      │      Custom Setup                                            │
      └──────────────────────────────────────────────────────────────┘

      To accept the selection, press ENTER.

      To change the selection, press the UP or DOWN ARROW key
      until the item you want is selected, and then press ENTER.
```

Select the Express option. DBLSPACE will display information about your disk and the amount of time the compression will require, as shown here:

```
Microsoft DoubleSpace Setup

     DoubleSpace is ready to compress drive C. This will take 16
     minutes.

     Note : During this process, DoubleSpace will restart your
            computer to enable the DoubleSpace device driver.


     To compress this drive, press C.
     To return to the previous screen, press ESC.
```

Press C to continue the compression. DBLSPACE will first run the CHKDSK command to ensure that your disk's file allocation tables contain valid entries. Next, DBLSPACE will make temporary modifications to your AUTOEXEC.BAT and will restart your system. After your system starts, DBLSPACE will continue its compression. Depending on the number and size of files on your disk, the amount of time the operation requires will vary from several minutes to possibly two hours or more. After the compression completes, DBLSPACE will run DEFRAG on the newly compressed disk. When DEFRAG completes, DBLSPACE will display a summary screen, as shown here:

```
Microsoft DoubleSpace Setup


      DoubleSpace has finished compressing drive C.

           Free space before compression:  32.4 MB
           Free space after compression:  178.7 MB
           Compression ratio:             2.0 to 1
           Total time to compress:        2 hours and 29 minutes.

      DoubleSpace has created a new drive H that contains 5.9 MB
      of uncompressed space. This space has been set aside for
      files that must remain uncompressed.

      To exit from DoubleSpace and restart your computer, press
      ENTER.
```

Note the storage capacity your disk now has available.

Press ENTER to restart your system. You can now use your compressed disk just as you previously used drive C.

DBLSPACE Creates a Compressed and Uncompressed Disk Volume

When you use DBLSPACE to compress your disk, DBLSPACE conceptually divides your disk into two parts. DBLSPACE creates a compressed volume that consumes most of your disk space, as well as an uncompressed volume.

The uncompressed volume serves two functions. First, some applications must store information in an uncompressed format. One of the best examples of such an application is the Windows permanent swap file. Second, and most important, DOS uses the uncompressed volume to boot. Here's how. When your system first starts, the uncompressed disk is actually your computer's drive C. Your compressed drive uses a different drive letter. During the startup process, DOS actually switches drive letters so your compressed disk becomes drive C. DOS then processes the CONFIG.SYS and AUTOEXEC.BAT files that reside on the compressed drive.

NOTE *Don't worry about updating your LASTDRIVE statement in CONFIG.SYS—DBLSPACE does it automatically.*

Which Files Reside on the Uncompressed Disk?

DOS places on your uncompressed disk the five files listed in Table 44-1.

Note the file named DBLSPACE.000. That file contains the data stored on your compressed disk. In other words, there really isn't a compressed disk. Instead it's an illusion. When DBLSPACE compressed the files on your disk, it stored them all inside of a second file, the file DBLSPACE.000. When you perform file operations, DOS hides the second file from you, making the file appear instead as a disk drive.

Filename	Contents
IO.SYS	The initial DOS boot code and input/output drivers
MSDOS.SYS	The MS-DOS operating system kernel
DBLSPACE.BIN	The compression software used to manage your compressed disk
DBLSPACE.000	The compressed disk's data
DBLSPACE.INI	DBLSPACE configuration and initialization specifics

Table 44-1. Files that DOS Stores on the Uncompressed Disk

To better understand how DOS uses the compressed and uncompressed disk, format a bootable floppy drive as shown here:

```
C:\> FORMAT  A:/S <ENTER>
```

When FORMAT completes, boot your system from the floppy drive. Next, perform a directory listing of drive C using the following command:

```
A:\> DIR C: /A:SH <ENTER>
```

The /A:SH directs DIR to list hidden and system files. In most cases, DIR will display the files shown here:

```
A:\> DIR C: /A:SH <ENTER>

 Volume in drive C has no label
 Volume Serial Number is 1979-4F71
 Directory of C:\

IO       SYS     39590 10-26-93   6:00a
MSDOS    SYS     37416 10-26-93   6:00a
DBLSPACE BIN     63868 10-26-93   6:00a
DBLSPACE INI        55 11-25-93  12:35p
DBLSPACE 000  54775089 11-25-93   2:27p
        5 file(s)   54916018 bytes
                     6631424 bytes free
```

As you can see, DIR displays the files stored on your "uncompressed disk." The rest of your files and directories are not visible. For DOS to access the "compressed disk," DOS must load the file DBLSPACE.BIN, which provides the software support that makes the file DBLSPACE.000 appear as a disk drive. Remove the floppy disk from drive A and restart your system.

The DBLSPACE.SYS Device Driver

If you examine your CONFIG.SYS file, you may find a DEVICEHIGH entry for the DBLSPACE.SYS device driver. If DOS encounters the DBLSPACE.SYS entry with a /MOVE switch, DOS will move the driver DBLSPACE.BIN (loaded into conventional memory at startup) to an upper memory block. If your CONFIG.SYS file does not contain a DBLSPACE.SYS entry, you will still have full access to your compressed data. DOS, however, will leave the driver DBLSPACE.BIN in conventional memory.

DBLSPACE and SMARTDRV

If you are using DBLSPACE, you should still use the SMARTDRV disk-caching software discussed in Chapter 31. SMARTDRV is fully compatible with DBLSPACE, and its use will significantly improve your system performance.

SUMMARY

Doubling Your Disk's Storage Capacity

DBLSPACE is a DOS 6 command that transparently compresses the files stored on your disk. As a result of the compression, most users may nearly double the storage capacity of their disks. To compress your disk, invoke DBLSPACE from the DOS prompt as shown here:

```
C:\> DBLSPACE <ENTER>
```

DBLSPACE will compress each of your existing files, which may require several minutes to a few hours.

Compressed and Uncompressed Drives

DBLSPACE creates two logical drives on your disk. One drive is compressed and one is not. The compressed drive contains your existing files and will be the drive you regularily use as drive C. Each time you create a file on the compressed drive, the file is automatically compressed. Should you later copy the file to an uncompressed drive, the file is stored in its uncompressed format. The uncompressed drive serves two purposes. First, DOS uses the drive to boot. During the system startup, DOS changes drive letters making your compressed drive appear as drive C. Second, some programs have files whose contents cannot be compressed. In such cases, you can store the file on the uncompressed drive.

SUMMARY

The DBLSPACE.SYS Device Driver

DOS provides the DBLSPACE.SYS device driver, which lets you move the DBLSPACE.BIN software from conventional memory to an upper memory block. The format of the CONFIG.SYS entry is as follows.

```
DEVICEHIGH=C:\DOS\DBLSPACE.SYS /MOVE
```

If the entry is not in your CONFIG.SYS file, the DOS 6 disk compression will still work. The only difference is that DOS will leave DBLSPACE.BIN in conventional memory.

DBLSPACE and SMARTDRV

If you are using DBLSPACE, you should still use the SMARTDRV disk-caching software discussed in Chapter 31. SMARTDRV is fully compatible with DBLSPACE and its use will significantly improve your system performance.

Hands On

If you are using DOS 6 and you have not yet invoked DBLSPACE to increase your disk's storage capacity, turn to Chapter 22 and perform a complete backup of your disk. When the backup completes, invoke DBLSPACE, as shown here:

```
C:\> DBLSPACE <ENTER>
```

Most users will want DBLSPACE to perform an express compression operation. Follow the remaining prompts as discussed earlier in this chapter.

Review

1. Why does DBLSPACE create two logical drives?
2. Is the compressed drive a unique disk partition?
3. How can you view the files on the uncompressed drive?
4. What happens when you copy a compressed file to a floppy disk?
5. What is the purpose of the DBLSPACE.SYS device driver?

Answers

1. DBLSPACE creates two logical drives on your disk. One drive is compressed and one is not. The compressed drive contains your existing files and will be the drive you regularily use as drive C. The uncompressed drive serves two purposes. First, DOS uses the drive to boot. During the system startup, DOS changes drive letters making your compressed drive appear as drive C. Second, some programs have files whose contents cannot be compressed. In such cases, you can store the file on the uncompressed drive.
2. The compressed drive is not a unique disk partition. Actually, it is a file named DBLSPACE.000. DOS hides the file on the uncompressed drive. During system startup, DOS loads software (DBLSPACE.BIN) that makes the file appear as a disk drive.
3. DBLSPACE lets you refer to the uncompressed drive using a specific disk drive letter. Assuming your uncompressed drive is drive G, you can list the files it contains as follows:

```
C:\> DIR  G: /S /A:SH <ENTER>
```

4. Each time you use a compressed file, DBLSPACE will decompress it. Therefore, if you copy a compressed file to a floppy, the floppy will contain the decompressed file.
5. The DBLSPACE.SYS device driver lets you move the DBLSPACE.BIN driver from conventional memory to upper memory. The DBLSPACE.SYS driver is not required to access the compressed drive.

Advanced Concepts

Although most users will use DBLSPACE to compress their disk and then forget about the command, DBLSPACE actually provides several command line switches that you may also want to use. This section examines several of these switches. The Command Reference at the end of this book also describes each of the DBLSPACE command line switches in detail.

Compressing a Floppy Drive

The DBLSPACE command lets you compress both hard and floppy disks. Because floppy disks are removable, the steps you must perform to compress and later use a floppy disk will differ from those for your hard disk. To begin, place the floppy disk that you want to compress in the floppy drive. Assuming the disk resides in drive A, you can compress the disk as follows:

```
C:\> DBLSPACE /COMPRESS A: /NEWDRIVE=F: <ENTER>
```

When DBLSPACE completes, you can refer to the floppy's compressed volume as drive A and the uncompressed volume as drive F. If you do not specify a letter for the uncompressed drive, DBLSPACE will use the next available drive letter.

Determining Available Drives

The previous DBLSPACE command used drive F for the floppy disk's uncompressed volume. To determine the drive letters that are available for use, invoke DBLSPACE using the /LIST switch as shown here:

```
C:\> DBLSPACE /LIST <ENTER>
```

Mounting and Unmounting Compressed Floppy Disks

After you compress a floppy disk, you may remove the floppy and insert a second one. Each time you insert a compressed floppy disk, you must "mount" the floppy for use. Mounting a floppy disk is really nothing more than the process of telling DOS the floppy contains a compressed volume and that you want to use it. The following DBLSPACE command mounts the compressed floppy disk contained in drive A.

```
C:\> DBLSPACE  /MOUNT  A: <ENTER>
```

Later, if you finish using the floppy, you can unmount the disk using the /UNMOUNT switch as shown here:

```
C:\> DBLSPACE  /UNMOUNT  A: <ENTER>
```

Defragmenting a Compressed Volume

As you've read, DOS represents the compressed drive using a file. Every file stored on the compressed drive resides in the (compressed volume) file. As DOS stores and deletes files in the compressed volume file, it is possible for fragmentation to occur. Because the compressed volume file does not use the standard DOS file format, you cannot use the DEFRAG command to correct it. Instead, you must invoke DBLSPACE using the /DEFRAG switch.

```
C:\> DBLSPACE /DEFRAG <ENTER>
```

The following DBLSPACE command defragments the compressed disk that resides in drive A:

```
C:\> DBLSPACE /DEFRAG A: <ENTER>
```

Checking the Compressed Disk's Status

Because DBLSPACE stores files in a unique format, the CHKDSK command cannot examine a compressed disk's internal format. To check the status of a compressed disk, invoke DBLSPACE using the /CHKDSK switch as shown here:

```
C:\> DBLSPACE  /CHKDSK <ENTER>
```

If DBLSPACE reports errors, you can correct them by including the /F switch as shown here:

```
C:\> DBLSPACE /CHKDSK /F <ENTER>
```

How DBLSPACE Compresses Data

As you have just read, DBLSPACE compresses and decompresses files on the fly as you use them. Microsoft refers to this process of compressing and decompressing files as the files are read and written as *real-time compression.* To perform its compression, DBLSPACE looks for repeated character strings or repeated patterns of 1's and 0's. Many files, for example, contain long sequences of 0's. Assume, for example, a file ends with 256 consecutive 0's. An uncompressed file would store each one of the zeros (0000...) consuming 256 bytes. A compressed file, however, would represent the 0's as a special symbol, one 0, and then a repeat count (R,0,256). In this case, the 256 bytes can be represented using only three bytes! Although this is a very simple discussion of compression, it illustrates how compression can save large amounts of disk space. The actual data-compression technique that DBLSPACE employs is based on the Lempel-Ziv data-compression algorithms.

As you read in Chapter 33, DOS stores files using disk clusters. One of the problems resulting from this is wasted disk space due to unused clusters. Because DBLSPACE stores files in its own format (within the file DBLSPACE.000), DBLSPACE is not restricted to using disk clusters in the same manner as DOS. As a result, DBLSPACE eliminates considerable wasted disk space due to unused clusters.

Key Terms

Real-time data compression The compression and decompression of data as it is written to or read from disk.

Uncompressed disk volume A disk volume whose contents are not compressed.

Compressed disk volume A logical disk volume whose contents are stored in a compressed format.

Chapter 45

The History of DOS

MS-DOS version 1.0 was originally released with the IBM PC in 1981. Since that time DOS has undergone many changes, becoming the most widely used operating system in the world. This chapter examines these changes in detail. If you are using DOS 6, you'll learn about many of the design decisions that led to DOS 6's development. If you are using a different version of DOS, you'll learn the differences between your DOS version and newer versions. If you really want to understand DOS, spend time reading this chapter. This information is presented at the end of this book because you now have the knowledge to understand fully and appreciate the discussion.

Before DOS

Although most users equate the term "PC" with the IBM PC or a PC-compatible originally introduced in 1981, the personal computer was actually in use throughout the 1970s. DOS is the operating system for the IBM PC and compatibles. Before DOS some computers used an interactive BASIC programming language for their operating system, while others used an operating system called CP/M, developed by Digital Research (the company who provides DR DOS).

Microsoft Corporation, the developers of MS-DOS, became a company in the late 1970s while they were developing programming languages such as BASIC for different microcomputers.

In 1980, IBM was designing the original IBM PC and asked Microsoft to develop several programming languages for their new computer. At this time CP/M was the operating system of choice and the leading candidate as the operating system for

IBM's emerging computer. As Microsoft progressed in their design of languages and utilities for the IBM PC, they recognized that producing an operating system for the PC was a natural extension of their efforts. Unfortunately, they did not have enough time to develop an operating system from scratch. As a solution, Microsoft purchased, from the nearby Seattle Computer Products, 86-DOS, an operating system for the 8086 processor, which was very similar to the 8088 processor in the original IBM PC.

DOS 1

Using the 86-DOS operating system as its base, Microsoft developed MS-DOS version 1.0, the operating system released with the original IBM PC in 1981. Table 45-1 lists the commands provided in DOS 1.

DOS 1.0 did not support tree-like (or hierarchical) directories. All file accesses where made through file control blocks, a technique borrowed from CP/M. DOS version 1.0 supported two types of executable files, COM and EXE. COM files used a format similar to CP/M executables and were restricted to 64Kb of memory for code, data, and stack space. In addition, version 1.0 introduced DOS batch files.

Because the original IBM PC typically shipped with only 64Kb to 256Kb, memory was a scare commodity. To reduce the amount of memory needed by the command processor (COMMAND.COM), version 1.0 divided the command processor into a resident section and a transient section whose contents DOS could overwrite during a program's execution. After each program ended, the resident section would examine the transient section, and if necessary, would load it from disk back into memory.

When Microsoft provided DOS to IBM, IBM used the name PC-DOS to illustrate the close relationship to their IBM PC.

CHKDSK	DEBUG	ERASE	REM
COMMAND	DIR	FORMAT	REN
COMP	DISKCOMP	LINK	SYS
COPY	DISKCOPY	MODE	TIME
DATE	EDLIN	PAUSE	TYPE

Table 45-1. Commands Provided in DOS 1

DOS 1.1

In 1982, the second release of DOS, called MS-DOS version 1.25 by Microsoft and PC-DOS version 1.1 by IBM, provided support for double-sided disk drives. The initial single-sided floppy drives shipped with the original IBM PC could only access information stored on one side of the disk at a time. To reduce the number of disks required, users could buy disks called *flippies.* They could store information on one side of the flippy disk, take the disk out of the drive and "flip" it over to store information on the other. In addition to the support for double-sided disks, DOS 1.1 corrected some errors found in version 1.0 and provided the programmer's tool EXE2BIN.

DOS 2

The original IBM PC was a floppy disk-based system, with floppy drives A and B. The original PC did not support hard disks, which, at the time, were very expensive. When IBM announced plans for the IBM PC XT, which contained a 10Mb hard disk, Microsoft had to develop a new file system for DOS. MS-DOS versions 1.0 and 1.1 did not support tree-like directories. For hard disks, however, such directories were a necessity.

Microsoft chose to develop a tree-like directory structure similar to the minicomputer UNIX operating system, whose popularity was beginning to grow at that time. Because file control blocks did not provide space for directory path names, Microsoft put file control blocks aside and based version 2.0 on file handles. A second benefit DOS obtained from file handles was the ability to perform I/O redirection (also a popular feature from UNIX).

By DOS version 2.0, the IBM PC was becoming very successful, and DOS was catching up with CP/M as the operating system of choice. Because of the PC's popularity, many hardware manufacturers developed PC-based products. To help these manufacturers integrate their products, MS-DOS 2.0 provided support for installable device drivers, providing the first CONFIG.SYS customization file. DOS version 2.0 contained many UNIX characteristics. UNIX, however, is a multiuser operating system that lets many users run several programs at one time.

To provide a very simple form of multitasking, version 2.0 provided support for memory-resident commands, such as GRAPHICS and PRINT.

Table 45-2 summarizes the commands added to DOS in version 2.

Following version 2.0, Microsoft released version MS-DOS 2.01, which provided support for international symbol sets. During this time, IBM was releasing its short-lived IBM PCjr computer. To support the PCjr, Microsoft developed PC-DOS 2.1 for IBM. Microsoft later combined these latest two DOS versions to produce MS-DOS 2.11. In late 1983, Microsoft released MS-DOS 2.25, which contained fixes to bugs and support for the extended ASCII character set.

ASSIGN
BACKUP
BREAK
CHDIR (CD)
CLS
CTTY
ECHO
EXIT
FC
FDISK
FIND
FOR
GOTO
GRAPHICS
IF
MKDIR (MD)
MORE
PATH
PRINT
PROMPT
RECOVER
RESTORE
RMDIR (RD)
SET
SHIFT
SORT
TREE
VER
VERIFY
VOL

Table 45-2. *Commands Introduced in DOS 2*

DOS 3

In 1984, IBM released its 80286 IBM PC AT, which used large 1.2Mb floppy drives and stored setup information about the computer in a CMOS chip. To support the PC AT, Microsoft released DOS version 3.0. The 1984 time period was the eve of computer networks. Although the widespread use of local area networks was still in the future, much of DOS 3.0 was written with the future support for networks in mind. Table 46-3 lists the commands added to DOS in version 3.0

DOS support for local area networks arrived with version 3.1. Although DOS had replaced file control blocks (FCBs) with file handles in DOS 2, programs still existed that used FCBs. To reduce the misuse of file control blocks in network programs, DOS 3.1 only let four file control blocks be open at one time. If a program opened a fifth, the network server or SHARE would close the oldest file control block. To provide support for additional file control blocks, DOS 3.1 added the CONFIG.SYS FCBS entry. In addition, DOS 3.1 introduced the JOIN and SUBST pretender commands.

In 1986, Microsoft released version 3.2, which provided support for 3 1/2-inch micro floppy disk drives. In addition, DOS 3.2 introduced the REPLACE and XCOPY commands.

ATTRIB
GRAFTABL
KEYB*xx*
LABEL
SELECT
SHARE

Table 45-3. *Commands Introduced in DOS 3*

In 1987, IBM released its PS/2 line of computers. To support the PS/2, Microsoft released DOS version 3.3. In addition to supporting the PS/2, DOS 3.3 introduced five commands: CALL, APPEND, KEYB CHCP, NLSFUNC, and FASTOPEN. DOS 3.3 was the most widely used and most popular version of DOS. In fact, many users today still run DOS 3.3 because they know it works well. The major shortcoming of version 3.3 is it only supports disk partitions up to 32Mb in size.

DOS 4

In 1988, DOS version 4.0 was released to address the 32Mb disk partition-size restriction. In version 4.0, disk partitions can contain up to 512Mb. In addition, DOS 4.0 provided a menu-driven shell program that let users select files using menus or by pointing at a program or file and clicking the mouse. DOS 4.0 also included the MEM command that let users not only display the contents of their computer's conventional memory, but also extended and expanded memory. In addition, version 4.0 updated several commands to make better use of memory.

The initial version of DOS 4.0 contained bugs and received poor reviews. To correct the bugs, Microsoft released version 4.01. Unfortunately, most users and many manufacturers had decided not to upgrade to DOS 4 by that time.

DOS 5

Beginning in 1987, the personal computer revolution became the local area network revolution. Offices everywhere across the United States began connecting personal computers together to share information. It did not take long before user programs became so large that they had difficulty running within the 640Kb conventional memory limits imposed by DOS.

In some cases, users made use of expanded and extended memory. Unfortunately, many programs, including DOS, could not run in memory addresses above 640Kb.

In 1990, Microsoft had tremendous success with Windows, Microsoft's user-friendly graphical interface. Under Windows, new users learned how to operate their computer faster, and experienced users became more productive by running several programs at the same time.

Also in 1990, third-party disk utilities erupted into a several hundred million dollar annual market, selling "the utilities DOS forgot."

In 1991, Microsoft released DOS version 5.0, the result of a decade of improvements to the original 86-DOS.

DOS version 5 addresses memory restrictions by not only providing enhanced, extended, and expanded memory support, but also the ability to run DOS in high memory and to load device drivers and memory-resident commands into reserved memory.

DOS 5 also included a much more powerful menu-driven shell to replace the DOS 4 predecessor. In addition, version 5 lets users quickly recall previous commands and define memory-resident macros, which behave like small fast batch files.

To provide disk utilities to all users, DOS 5 provides commands that undelete files and rebuild a disk after an accidental disk format operation.

Last, to keep pace with the growth in hard disk sizes, DOS 5 supports disk partitions up to 2 gigabytes!

Table 45-4 lists the commands introduced in version 5.

DOS 6

In early 1993, Microsoft released DOS 6, which continued the DOS 5 trend of providing the user with utilities they had to previously purchase from third-party software companies. First, DOS 6 provides the INTERLNK and INTERSVR utilities, which let laptop computers easily exchange files with a desktop PC. In the future, many newer computers will contain power management chips that control the computer's power usage, so DOS 6 provides the POWER command that lets DOS control the chip (and hence the computer's power) in systems that support APM (advanced power management). DOS 6 also provides a collection of virus-protection programs and a utility to defragment your disk. To help users customize their system, DOS 6 introduced CONFIG.SYS entries that let users create menu-based configurations they can easily select when the system starts. The CONFIG.SYS processing also lets the user "single-step" through each CONFIG.SYS entry, directing DOS to individually install or ignore each entry.

Just prior to the release of DOS 6, Microsoft released Windows for Workgroups, a network-based version of Windows, which allows users to share data, send and receive electronic mail, and use remote printers. To help network-based DOS users take advantage of these features without having to run Windows for Workgroups, DOS 6 provided the NET command. Finally, DOS 6 provides a powerful menu-driven backup utility and disk-compression software that allows you to essentially double the storage capacity of your hard disk. Table 45-5 presents the commands introduced in DOS 6.

DOSKEY
EDIT
EXPAND
HELP
LOADFIX
LOADHIGH
MIRROR
SETVER
UNDELETE
UNFORMAT

Table 45-4. Commands Introduced in DOS 5

CHOICE
DBLSPACE
DECOMP
DEFRAG
DELTREE
FASTHELP
INTERLNK
INTERSVR
MEMMAKER
MOVE
MSAV
MSBACKUP
MSD
POWER
SMARTDRV
VSAFE

Table 45-5. *Commands Introduced in DOS 6*

Conclusion

As you can see, DOS has evolved to support new computers, new hardware, and the ever-changing demands of users. It is impossible to expect one operating system to satisfy everyone's needs. DOS, however, continues to meet the demands of more than 90 million users and will continue to do so for some time. Just as DOS has come a long way since 1981, you have come a long way since you began Chapter 1. Congratulations on your journey, and may DOS be good to you.

Kris Jamsa

Command Reference

APPEND

Defines a data-file search path

When DOS cannot find a data file as specified (or in the current directory), it searches to see whether the user has defined a data-file search path (a sequence of directories DOS can follow to carry out a command). The APPEND command allows you to define disk drives and subdirectories to be included in this path. Hence, this command defines a *data-file search path* that DOS uses whenever it fails to locate a file in either the current directory or a specified directory.

Format

```
APPEND[d:][p][;[d:][p]...]
APPEND[/X][/E]
APPEND[;]
```

or, in DOS 5 or later:

```
APPEND[/X:on|off][/PATH:on|off]
```

D: A disk drive to include in the data-file search path.

P A subdirectory to include in the data-file search path.

/X Aids in SEARCH-FIRST, FIND-FIRST, and EXEC options.

/E Places an APPEND entry in the DOS environment in a manner similar to a PATH entry (see PATH).

/X:on|off A DOS 5 or later switch that works like the /X parameter in earlier DOS versions. When /X:on is specified, the application can look for data files in the appended directories. When you specify /X:off, the application can look only in the current directory. You can abbreviate /X:on as simply /X.

/PATH:on|off A DOS 5 or later switch that specifies how applications use the data-file search path when looking for files with disk drive or directory information specified. The default setting is /PATH:on. This allows searching the appended directories even if a drive or directory is specified in the request. If you specify /PATH:off, the appended directories are searched only if the request does not include a drive or directory.

Notes

To reduce overhead with each file reference, place the directories most likely to contain data files at the beginning of the APPEND path. When DOS cannot locate a file in the current directory, it searches the first entry in the APPEND path; if it can't locate the file in that directory, it searches the second, third, and so on. If commonly used data files reside in the directories specified at the end of the APPEND path, DOS has to perform needless directory searches.

Not all commands use the data-file search path by default. To increase the number of commands that use it, use the /X parameter.

If you invoke APPEND with no command-line parameters, APPEND displays the current data-file search path.

To remove the current data-file search path, invoke APPEND followed by a semicolon (see the preceding "Format" section).

APPEND supports network drives and path names. Do not use APPEND with Microsoft Windows.

Examples

In the following example, if DOS cannot find the data file in the current directory, it searches the root directories on drives C, B, and A, in that order:

```
C:\> APPEND C:\;B:\;A:\ <ENTER>
```

The following APPEND command tells DOS to search the directories \DOS, \UTIL, and \MISC, all of which reside on drive C:

```
C:\> APPEND C:\DOS;C:\UTIL;C:\MISC <ENTER>
```

Messages

APPEND already installed

The APPEND command has already been invoked. Once invoked, APPEND installs memory-resident software that handles processing of the data-file search path. To modify the data-file search path, omit a disk drive specifier or subdirectory path that may precede APPEND in the command line.

APPEND/ASSIGN Conflict

You have issued the ASSIGN command before issuing the APPEND command. You must remove any disk drive reassignments before invoking APPEND.

No Append

The data-file search path is not currently in use.

ASSIGN

Routes disk drive references from one disk drive to another

Many older software packages expect to find data or overlay files on drive A. If you want to install such software on a fixed disk, you must trick the software into looking for the files on the fixed disk. ASSIGN lets you do this by routing disk drive references from one disk drive to another. DOS 6 does not support the ASSIGN command.

Format

```
ASSIGN[source_drive=target_drive[...]]
```

or, in DOS 5:

```
ASSIGN [/STATUS]
```

source_drive The disk drive identifier of the disk from which I/O references are routed.

target_drive The disk drive identifier of the disk to which I/O operations are routed.

/STATUS A DOS 5 parameter that displays the currently active drive assignments. This parameter may be abbreviated /S or /STA.

Notes

Invoking ASSIGN without command-line parameters restores the default disk drive assignments.

Do not place a colon after each disk drive identifier in the ASSIGN command line.

Do not use an ASSIGNed disk with the BACKUP, DISKCOMP, DISKCOPY, FORMAT, JOIN, LABEL, PRINT, RESTORE, or SUBST commands.

Most users should consider using the SUBST command instead of ASSIGN. Microsoft recommends this to ensure compatibility with future versions of DOS. The following ASSIGN and SUBST commands are functionally equivalent:

```
C:\> ASSIGN A=C <ENTER>
C:\> SUBST A: C:\ <ENTER>
```

Examples

In the following example, DOS disk I/O operations, which reference drive A, are routed to the disk in drive C:

```
C:\> ASSIGN A=C <ENTER>
```

If you then issue a command such as DIR A:, DOS will list the files on drive C.

You can perform multiple disk drive assignments on one command line as follows:

```
C:\> ASSIGN A=C B=C <ENTER>
```

ATTRIB

Displays or modifies a file's attribute byte

This command displays or modifies a file's attribute byte, which contains information about the file. Several DOS commands, such as BACKUP, RESTORE, and XCOPY, use a file's attribute to enable selective file processing. By using these commands in conjunction with ATTRIB, you can gain considerable file-processing control.

Format

```
ATTRIB[+A|-A][+R|-R]file_spec[/S]
```

or, in DOS 5 or later:

```
ATTRIB[+A|-A][+R|-R][+S|-S][+H|-H]file_spec[/S]
```

+A Sets a file's archive bit.

–A Clears a file's archive bit.

+R Sets a file's read-only bit.

–R Clears a file's read-only bit.

+S Sets the file's system bit (DOS 5 or later).

–S Clears the file's system bit (DOS 5 or later).

+H Sets the file's hidden bit (DOS 5 or later).

–H Clears the file's hidden bit (DOS 5 or later).

file_spec The complete DOS file specification, including the disk drive and path name, of the file(s) to be modified. ATTRIB supports DOS wildcard characters.

/S Tells DOS to process all files in the subdirectories below the specified directory (DOS 3.3 and later).

Notes

Every file on the disk has specific characteristics or attributes. The ATTRIB command lets you change a file's attributes. Users most commonly change the read-only and archive attributes. A file with the read-only attribute set cannot be overwritten or deleted.

The archive attribute helps DOS determine which files on the disk need to be backed up. Whenever you create a file or change an existing file's contents, DOS sets the file's archive attribute. The BACKUP command searches the disk for files whose archive attribute is set and then, having copied the files to a backup disk, clears the file's archive bits.

In DOS 5 or later, you can change a file's system and hidden bits. The system attribute is normally reserved for the files DOS uses whenever your system is started. A file whose hidden bit is set will not appear in a directory listing.

Examples

In the following example, ATTRIB sets the CONFIG.SYS file to read-only:

```
C:\> ATTRIB +R CONFIG.SYS <ENTER>
```

To show the current attributes of all files in the current directory, use the following:

```
C:\> ATTRIB *.* <ENTER>
```

In DOS 5 or later, the following command hides the file MY_NOTES.TXT:

```
C:\> ATTRIB +H MY_NOTES.TXT <ENTER>
```

Messages

Parameter format not correct

You have specified an invalid parameter, or used multiple parameters without intervening spaces.

Syntax error

The command line contains an option other than +A, –A, +H, –H, +R, –R, +S, or –S.

BACKUP

Backs up one or more files to a new disk

The BACKUP command works closely with the operating system's directory entries to select specific files for backup. Note that DOS directory fields (information about each file in the directory) relate closely to parameters included in the BACKUP command line (see the "Format" section for a complete list of these parameters). In DOS 6, BACKUP has been replaced by the MSBACKUP command (see MSBACKUP).

Format

```
BACKUP source:[file_spec]target:[/A]
  [/D:mm-dd-yy][/L:log_file][/M][/S]
  [T:hh:mm:ss][/F[:Size]]
```

source: Specifies the source disk that contains the file(s) to be backed up.

file_spec The path name(s) for the file(s) to back up.

target: Specifies the target disk.

/A Appends source files to files on the target disk.

/D:mm-dd-yy Backs up files modified since this date.

/L:log_file Places an entry for all files in the file specified by log_file. BACKUP.LOG is the default.

/M Backs up files modified since the last backup.

/S Backs up all subdirectory files.

/T:hh:mm:ss Backs up files modified since the specified time (DOS 3.3 or later).

/F Tells DOS to format an unformatted disk (DOS 3.3 or later). If you are using DOS 5, you can specify the size of the floppy disk BACKUP is to format; use /F:size, where size is one of the following:

160	160K	160Kb
180	180K	180Kb
320	320K	320Kb
360	360K	360Kb

720	720K	720Kb			
1200	1200K	1200Kb	1.2	1.2M	1.2Mb
1440	1440K	1440Kb	1.44	1.44M	1.44Mb
2880	2880K	2880Kb	2.88	2.88M	2.88Mb

Notes

Whenever you use your computer, back up your files. If you are using a fixed disk, backups are easy. Depending on the size of the disk and the number of files it contains, backups may take considerable time. Make sure that you have enough formatted floppy disks. Use the following formulas to calculate how many floppy disks you need:

Bytes in use = Total disk space – Bytes available

Disks needed = Bytes in use ÷ Floppy disk capacity

The BACKUP command completes with one of the following exit status values:

Exit Status	**Meaning**
0	Successful backup
1	No files found to back up
2	Sharing conflicts prevented some files from being backed up
3	User termination via CTRL-C
4	An error prevented system backup

Examples

The following example backs up all files on drive C to the disk in drive A:

```
C:\> BACKUP C:\*.* A: /S <ENTER>
```

The following example uses the /A parameter to add C:TEST.DAT to the files contained on the backup disk in drive A:

```
C:\> BACKUP C:TEST.DAT A: /A <ENTER>
```

The following example backs up only those files created since April 30, 1993:

```
C:\> BACKUP C:\*.* A: /S /D:4-30-93 <ENTER>
```

The following command performs an incremental backup operation:

```
C:\> BACKUP C:\*.* A: /S /A /M <ENTER>
```

Messages

***** Backing up files to drive *n*: *****
Diskette Number: *n*

Indicates the disk drive to which BACKUP is backing up files, as well as the disk number it is creating. Use the disk number when labeling your backup disks.

Fixed backup device *n* is full

The target drive for BACKUP is a fixed disk and the disk is full.

Insert backup diskette *n* in drive *n*:
Strike any key when ready

BACKUP is prompting you to insert a specific backup disk in the target drive. After you have done so, press any key to continue BACKUP.

Insert last backup diskette in drive *n*:
Strike any key when ready

The BACKUP command line contains the /A parameter, which tells BACKUP to add specific files to the backup disk in the target drive. In this case, BACKUP is prompting you to place the last disk used in a backup operation into the specified drive. Then press any key to continue.

Last backup diskette not inserted
Insert last backup diskette in drive *n*:
Strike any key when ready

The disk you inserted in the target drive is not the last disk you used for a BACKUP operation. Insert the correct disk and press any key to continue the backup.

***** Last file not backed up *****

BACKUP did not back up the last file displayed on screen. This error, which normally occurs when the backup target disk becomes full, occurs only when you use a fixed disk as the target disk.

No such file or directory

Either the filename or path name you specified as the source of the files does not exist. Double-check your spelling and make sure that you are specifying a complete path name, starting at the root directory, if necessary.

Source and target drives are the same

You have specified the same disk drive as both the source and target disk drives. Reissue the BACKUP command, specifying different disk drives.

Warning! Files in the target drive
***n*:\root directory will be erased**
Strike any key when ready

If you continue with the BACKUP operation, all files in the target disk's root directory will be erased. Since you should be using newly formatted disks, this message should be purely informational.

Warning! No files were found to back up

Either no files requiring a backup were found or no files matched the file specifications in the BACKUP command line.

BREAK

Enables or disables DOS extended CTRL-BREAK checking

By default, after completing keyboard, screen, and printer I/O operations, DOS checks for a user-entered CTRL-BREAK. If you enable extended CTRL-BREAK checking by entering a BREAK=ON command, DOS also checks for a CTRL-BREAK after completing services performed by the operating system, such as disk read or write operations.

Format

```
BREAK [ON|OFF]
```

ON Enables extended CTRL-BREAK checking.

OFF Disables extended CTRL-BREAK checking.

Notes

Because DOS must check for a CTRL-BREAK upon completing each system service, by using the BREAK command you increase the amount of time the system uses to complete operations. Programmers may want to set BREAK to ON during program development, but most users will leave the setting at the default BREAK=OFF.

Invoked without a command-line parameter, BREAK will display the current state of checking (either ON or OFF).

By default, DOS boots with extended CTRL-BREAK checking turned off. To enable extended checking at system startup, place the entry

```
BREAK=ON
```

in the file CONFIG.SYS and reboot.

Example

The following command enables DOS extended CTRL-BREAK checking:

```
C:\> BREAK ON <ENTER>
```

Message

Must specify ON or OFF

When you invoked the BREAK command, you specified a command-line parameter other than ON or OFF. Reissue the command, either with no command-line parameters or with ON or OFF.

CALL

Invokes a batch file from within another batch file

DOS has difficulty invoking one batch file from within another. This is especially true when the invocation of a procedure appears in the middle of the batch file. If you must invoke a batch procedure from within a batch file, the CALL command enables you to do so.

Format

```
CALL BatchFile[argument[...]]
```

BatchFile	The name of the second batch file to invoke.
argument	A command-line parameter for the second batch file.

Notes

The CALL command requires DOS 3.3 or later.

This command is similar to the function provided by using COMMAND /C within a batch file (see COMMAND).

Example

You can include CALL in one batch file in order to invoke another, as in the following example:

```
CLS
CALL MYPROC
DATE
```

CHCP

Displays or changes the current code page

Whenever DOS displays a character on screen, it must first map the character's ASCII value to a specific letter in a chosen character set. DOS uses code pages to map characters to letters.

Format

```
CHCP[code_page]
```

code_page Specifies the desired code page. This parameter must have been previously prepared by the system as either the primary or secondary code page in CONFIG.SYS. Valid code-page entries include the following:

Code Page	Country
437	United States
850	Multilingual
860	Portuguese
852	Slavic (DOS 5 or later)
863	French Canadian
865	Nordic

Notes

CHCP, which requires DOS 3.3 or later, allows you to set temporarily the desired code page for a device. If you intend to use the specified code page regularly, use the COUNTRY= entry in CONFIG.SYS rather than CHCP.

DOS allows you to use different character sets, thus offering international character support. To select an alternative code page (that is, a special character set), you previously must have issued the NLSFUNC command (see NLSFUNC). By entering a code-page value with the CHCP command, you reassign an alternative character set to be used by DOS.

When invoked without a command-line parameter, CHCP displays the current code page.

Example

The following command tells CHCP to select the Nordic code page:

```
C:\> CHCP 865 <ENTER>
```

Messages

Code page *nnnn* not prepared for system

Make sure that you have installed the PRINTER.SYS and DISPLAY.SYS device drivers in CONFIG.SYS. Next, make sure that you have issued the NLSFUNC command and prepared the code page for use by the MODE command.

File not found

CHCP could not find the file COUNTRY.SYS. Make sure that COUNTRY.SYS resides in the current directory, or that you specify the location of this file when you invoke NLSFUNC.

CHDIR

Changes or displays the default directory

To organize your files, DOS lets you group related files into directories. The CHDIR command changes or displays the current directory name for a specified disk drive. CHDIR can be abbreviated as CD.

Format

```
CHDIR[drive:][path]
CD[drive:][path]
```

drive: Specifies as the default the disk drive that contains the directory you want. If you omit this parameter, CHDIR uses the default drive.

path Specifies the path name for the current directory you want. If you omit the path, CHDIR will display the name of the current directory.

Notes

Whenever you specify a path name for CHDIR, the command does the following: If a slash precedes the path name (as in \SUBDIR), the search for the directory begins at the root directory. If the path name does not begin with a slash (as in SUBDIR), the search begins at the current directory. If you don't include a directory path, CHDIR displays the current directory.

Examples

Using CD without a path name displays the name of the current directory:

```
C:\> CD <ENTER>
C:\SUBDIR
```

This command is valid with a disk drive specifier, as shown here:

```
C:\> CHDIR A: <ENTER>
A:\UTIL
```

Likewise, the following command selects the subdirectory BATCH as the default directory:

```
C:\> CHDIR \BATCH <ENTER>
```

Message

Invalid directory

The subdirectory (or a path in the directory) you specified in the command does not exist. Check your spelling, making sure to include the necessary backslashes.

CHKDSK

Checks a disk's current status

CHKDSK examines the disk's file allocation tables and directory structures and reports errors and inconsistencies, providing you with the opportunity to correct the errors.

Format

```
CHKDSK [drive:][path][file_name][/F][/V]
```

drive: The disk drive CHKDSK is to examine.

path Specifies a subdirectory that contains the files CHKDSK is to examine for disk fragmentation.

file_name The filename and extension of the file(s) CHKDSK is to examine for disk fragmentation.

/F Fixes errors found in a directory or file allocation table.

/V Displays the names of all files on the disk.

Notes

CHKDSK reports on the status of the following disk conditions:

- The amount of free, used, and corrupted disk space
- The number of hidden files
- The amount of free and used memory

Occasionally, as normal day-to-day operations of a disk cause wear and tear on the storage media, files become corrupted and lose sectors. CHKDSK allows you to view and even repair such problems.

A file can also become fragmented, with its contents dispersed to different locations on the disk. CHKDSK also displays information on fragmented files.

CHKDSK does not work with joined or substituted disks (see JOIN and SUBST).

If you are operating on a local area network, do not try to repair the disk. When DOS has files open (as may be the case with a network), it cannot correctly update the file allocation table.

By default, CHKDSK only reports disk errors—it does not attempt to fix them. To write actual corrections to disk, use the /F parameter.

At least once a month, you should issue the CHKDSK command.

```
C:\> CHKDSK /F <ENTER>
```

to examine the fixed disk. DOS sometimes finds errors when it writes information to disk. Depending on the error, you may not realize that one has occurred. The CHKDSK command is your only tool for testing the "health" of the disk.

If CHKDSK finds lost information caused by an error, it displays the following:

```
nnn lost clusters found in n chains.
Convert lost chains to files  (Y/N)?
```

If you respond with **Y**, CHKDSK creates a file(s) containing the information referenced by the lost pointers. This occurs only if you have used the /F parameter. These files are placed in the root directory and named FILE0000.CHK, FILE0001.CHK, and so on. Use the TYPE command to examine the contents of these files. If they don't contain useful information, simply delete them.

Do not use the command CHKDSK /F from within Microsoft Windows or from within the MS-DOS shell when task swapping is enabled. Doing so might damage files stored on your disk.

Examples

The following command displays the state of the current disk:

```
C:\> CHKDSK <ENTER>
```

If you include a file specification or wildcard character, as follows:

```
C:\> CHKDSK *.* <ENTER>
```

CHKDSK reports on file fragmentation for the current directory.

To direct CHKDSK to repair any errors it may find, use the /F parameter, as shown in the following:

```
C:\> CHKDSK /F <ENTER>
```

If you invoke CHKDSK with the /V parameter:

```
C:\> CHKDSK /V <ENTER>
```

CHKDSK displays the name of every file on the disk.

Messages

Allocation error, size adjusted

The size of the file specified in a directory listing is inconsistent with the number of bytes CHKDSK found in the file allocation table. CHKDSK has corrected the error.

All specified file(s) are contiguous

The CHKDSK command line included a file specification CHKDSK is to examine for fragmentation. In this case, all the files were contiguous.

Cannot CHDIR to *pathname*
tree past this point not processed

CHKDSK cannot continue processing the subdirectory because of disk errors. Copy as many files as you can from the disk to backup disks. This error message is associated with a damaged disk.

Cannot CHDIR to root

CHKDSK cannot select the root directory as the current directory because of a disk error. Copy as many files as you can from the disk to backup disks. This error message is associated with a severely damaged disk.

CHDIR .. failed, trying alternate method

CHKDSK could not select the parent directory as the current directory by using the .. abbreviation. CHKDSK is trying an alternate method of selecting the parent directory. If the disk contains serious errors, CHKDSK will not be able to continue processing.

Directory is joined

CHKDSK cannot process the subdirectory specified because the directory is joined to another DOS disk. CHKDSK cannot process a subdirectory that is affected by the JOIN command.

Directory is totally empty, no . or ..

The subdirectory specified is probably damaged. In most cases, this error message is an indication of a severely damaged disk.

Disk error reading fat *n*

CHKDSK cannot read the file allocation table (FAT) specified.

Disk error writing fat *n*

CHKDSK cannot write to the file allocation table (FAT) specified.

. Does not exist

The abbreviated name for the current directory does not exist. If this error occurs, you should consider making a complete backup of all files on the disk. This may be a sign of disk errors to come.

.. Does not exist

The abbreviated name for the parent directory does not exist. If this error occurs, you should consider making a complete backup of all files on the disk. This may be a sign of disk errors to come.

Errors found, F parameter not specified
Corrections will not be written to disk

CHKDSK found errors while examining the disk specified. Since you did not place the /F parameter in the CHKDSK command line, CHKDSK will not record its corrections to disk. In most cases, you will want to reissue CHKDSK with the /F parameter.

File allocation table bad drive *n*:

CHKDSK cannot access the file allocation table on the disk drive specified. This error message is associated with severe disk errors.

***FILENAME.EXT* contains *n* contiguous blocks**

The CHKDSK command line included a file specification that CHKDSK is to examine for fragmentation. In this case, CHKDSK is displaying the number of clusters in a file that are not contiguous.

FILENAME.EXT* is cross-linked on cluster *n

DOS has erroneously assigned two files to point to the same cluster (set of sectors) on disk. Copy both of the files to a new disk and delete the originals. One of the files will be damaged (missing information). If the file is a text file, you will have to replace the missing data. If the file contains a program, do not execute it. Executing a damaged program can damage your disks and the information they contain.

First cluster number is invalid, entry truncated

The file specified contains an invalid cluster number as the first entry in the file allocation table. CHKDSK has truncated the file to 0 bytes.

General Failure error reading drive *n*:

The disk in the specified drive has probably not been formatted or is severely damaged.

Has invalid cluster, file truncated

CHKDSK has found an invalid cluster in the file allocation table entry for the file specified. As a result, CHKDSK has truncated the size of the file in question to the last correct entry.

***n* lost clusters found in *n* chains**
Convert lost chains to files (Y/N)?

The disk's file allocation table has clusters to files that no longer exist. CHKDSK is asking you whether you want to convert the lost clusters to files in the root directory. If the CHKDSK command line did not contain the /F parameter, you will have to reinvoke CHKDSK and specify /F if you want the lost pointers to be written to disk.

Unrecoverable error in directory
Convert directory to a file (Y/N)?

CHKDSK has encountered an error in the directory specified; this error prevents CHKDSK from continuing its processing. CHKDSK wants to know whether to convert the damaged directory to a file. In some cases, you may simply want to delete the damaged directory from the disk. If so, type **Y** and press ENTER; otherwise, type **N**.

CHOICE

Displays a user prompt and returns an error level that corresponds to a specific key pressed by the user.

Format

```
CHOICE  [/C[:]Keys [/N] [/S] [/T[:]Seconds,Default]
        [UserPrompt]
```

/C[:]Keys Specifies the letters the user can press. If you don't specify /C, CHOICE uses the letters YN as the default.

/N Directs CHOICE not to display the valid letters in the user prompt.

/S Directs CHOICE to be case sensitive. If you don't specify /S, CHOICE considers upper- and lowercase letters the same.

/T[:]Seconds,Default Specifies the number of seconds, from 0 through 99, that CHOICE will wait for a user response and the default selection should the time limit expire.

UserPrompt A character string that contains the prompt CHOICE displays to the user.

Notes

When the user presses one of the valid keys, CHOICE returns an exit status value that corresponds to the key. If the valid keys are ABC, A corresponds to the exit status 1, B to 2, and C to 3.

If the user presses a key that is not one of the specified options, CHOICE will beep the computer's built-in speaker. If the user presses CTRL-C to cancel the command, the error level 0 is returned.

CHOICE requires DOS 6.

Examples

The following batch file commands use CHOICE to decide whether or not the user wants to run CHKDSK:

```
@ECHO OFF
CHOICE  Run CHKDSK
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 CHKDSK
```

When you invoke this batch file, CHOICE will display the following:

```
Run CHKDSK[Y,N]?
```

The following commands illustrate how you can use CHOICE within a menu-based batch file:

```
@ECHO OFF
ECHO  A  Run WordPerfect
ECHO  B  Run Lotus 1-2-3
ECHO  C  Run Windows
ECHO  D  Quit
CHOICE /C:ABCD /N Selection:
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 WP
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 123
REM Other commands here
```

Finally, the following command gives the user ten seconds to respond to a yes or no prompt about whether to defragment the disk. If the user fails to respond, the default is N:

```
CHOICE  /T:10,N  Run  DEFRAG
```

Message

CHOICE: Incorrect *xxxx* syntax

One of the parameter values specified in your command line is invalid. See the CHOICE command format and valid parameter settings.

CLS

Clears the screen display

The CLS command clears the screen display and places the cursor (and DOS prompt) in the home (upper-left) position.

Format

```
CLS
```

Notes

This command does not affect the way DOS displays information on screen; it merely clears the screen of all information.

COMMAND

Loads a secondary command processor

COMMAND is used most often to invoke nested batch files in DOS versions earlier than 3.3.

Format

```
COMMAND [drive:][path][Ctty_Device]
    [/C string][/E:num_bytes][/K:Filename.Ext][/P][/MSG]
```

drive: Specifies the disk drive that contains the secondary command processor. If you do not specify a drive, DOS uses the default drive.

path The path name of the subdirectory that contains the command processor. If you do not specify a path name, DOS uses the default directory.

Ctty_Device A DOS 5 parameter that lets you specify an alternative device for I/O operations. For more information on changing this device, refer to the CTTY command.

/C string Tells DOS to execute the command specified by string. Ordinarily, this parameter is used for nested batch-file invocations. If this parameter is used with /P, DOS ignores /P.

/E:num_bytes Specifies the size of the area DOS is to allocate for the secondary command processor's environment space. This parameter must be between 160 and 32,767 bytes; the default is 160 bytes.

/K:Filename.Ext A DOS 6 switch that specifies a program or batch file to run before displaying the DOS prompt.

/P Tells DOS to leave the secondary command processor permanently in memory.

/MSG Directs DOS to store error messages in memory, for floppy disk-based systems.

Notes

COMMAND is commonly used to invoke nested batch procedures, as shown here:

```
CLS
COMMAND /C BATFILE
DATE
```

For information about terminating a secondary command processor, see EXIT.

If you are using DOS 3.3 or later, use the CALL command to invoke your nested batch files.

The /K switch provides a convenient way for you to run a specific batch file each time you create a DOS window from within Microsoft Windows. For example, if you have a batch file named SET_ENV.BAT that defines your system prompt and other environment settings, you can use the Windows PIF Editor to edit the file DOSPRMPT.PIF by adding the /K switch, /K:SET_ENV.BAT, to the optional parameters field.

Example

Using COMMAND, the following batch file, NESTED.BAT, invokes TIMEDATE.BAT:

```
@ECHO OFF
CLS
VER
COMMAND /C TIMEDATE
VOL
```

Messages

Bad command or file name

The string specified with the /C parameter is valid.

Bad or missing command interpreter

DOS could not locate COMMAND.COM in the root directory or in the directory specified in the SHELL= entry in CONFIG.SYS.

Invalid environment size specified

The COMMAND command line contains the /E parameter, which allows you to specify an environment size from 160 to 32,768 bytes. In this case, the value you have specified is invalid.

Specified COMMAND search directory bad

The subdirectory path specified as a parameter to COMMAND is invalid. Use the EXIT command to terminate the command processor; then reissue COMMAND with the correct directory path.

COMP

Compares the contents of two files

COMP compares the contents of two files and displays the first ten differences between the files as hexadecimal offsets into the file. DOS 6 does not support the COMP command.

Format

```
COMP file_spec1 file_spec2
```

or, using DOS 5:

```
COMP file_spec1 file_spec2 [/D]
  [/A][/L][/N=number][/C]
```

file_spec1 and file_spec2 Completes DOS path names of the files to be compared. COMP supports DOS wildcard characters.

/D Displays any differences in decimal (DOS 5 only).

/A Displays any differences in ASCII (DOS 5 only).

/L Displays the line numbers in which differences occur, instead of displaying offsets (DOS 5 only).

/N=number Compares only the number of lines specified (DOS 5 only).

/C Treats upper- and lowercase letters identically (DOS 5 only).

Notes

COMP compares two files and displays the differences as hexadecimal values. If the files are identical, COMP displays the message:

```
Files compare OK
```

After files have been compared, COMP asks:

```
Compare more files (Y/N)?
```

To compare additional files, press **Y**; otherwise, press **N**. If COMP locates more than ten differences, it displays

```
10 Mismatches - ending compare
```

COMP will not compare files of different sizes.

If you are copying a critical file from one disk or directory to another, you can verify that the copy was successful by using the COMP command after the copy procedure is complete. If differences exist between the two files, you know that an error occurred. If no differences exist, the copy was successful.

Examples

In the following example, COMP compares the contents of the file A.DAT to those of B.DAT:

```
C:\> COMP A.DAT B.DAT <ENTER>
```

Assuming that each file contains the following:

```
C:\> TYPE A.DAT <ENTER>
A
AA

C:\> TYPE B.DAT <ENTER>
B
BB
```

the command displays this:

```
Compare file C:A.DAT and file C:B.DAT
Compare error at OFFSET 0
File 1 = 41
File 2 = 42
Compare error at OFFSET 3
File 1 = 41
File 2 = 42
Compare error at OFFSET 4
File 1 = 41
File 2 = 42
Compare more files (Y/N)?
```

If you are using DOS 5, you can the display the differences between each file in ASCII, as follows:

```
C:\> COMP A B /A <ENTER>
```

If you omit the filename from the secondary file, DOS matches the filename on the specified drive to the primary file, as shown in the next example.

```
C:\> COMP A.DAT B: <ENTER>
```

If you do not specify a file in the COMP command line, COMP prompts you for it, as shown here:

```
C:\> COMP <ENTER>
Enter the primary file name.
A.DAT

Enter the 2nd file name or drive id.
B.DAT
```

Messages

10 Mismatches - ending compare

COMP has identified ten differences between the files and is terminating the comparison. If you are using DOS 3.3, the FC command lets you display all differences between two files.

Access denied

One of the files specified in the COMP command line is locked by another network program, which prevents COMP from accessing the file. Wait a few minutes and reissue the command.

Compare error at offset *nnnnnnn*
File1=*nn*
File2=*nn*

COMP has located a difference between the two files and is displaying the offset location of the difference and the byte values that differ. Remember: COMP displays these values in hexadecimal.

Compare more files (Y/N)?

COMP has completed one file comparison and is asking whether to perform another. If you want to, type Y; COMP will prompt you for the files to compare. If you type N, COMP terminates, returning control to DOS.

Enter 2nd file name or drive id

The COMP command line did not specify the second file COMP is to compare. In this case, simply enter the filename or disk drive. If you enter a disk drive letter, COMP searches that drive for a file that matches the primary filename.

Enter primary file name

The COMP command line did not specify the files COMP is to compare. COMP prompts you for them. In this case, simply enter the desired filename.

EOF mark not found

The last character in one of the files examined was not an end-of-file character (CTRL-Z). DOS uses the CTRL-Z character as the end-of-file marker for most text files. In most cases, this message is purely informational.

Files are different sizes

The files specified in the COMP command line differ in size. COMP will not compare files of different sizes. If you are using MS-DOS 3.3, use the FC command.

File sharing conflict

COMP could not access one of the files specified in the COMP command line because of network file sharing. Wait a few minutes for the file to be released and then reissue the COMP command.

Name of first file to compare:

The COMP command line did not specify the files COMP is to compare. COMP prompts you for them. In this case, simply enter the desired filename.

Name of second file to compare:

The COMP command line did not specify the second file COMP is to compare. In this case, simply enter the filename or disk drive. If you enter a disk drive letter, COMP searches that drive for a file that matches the primary filename.

COPY

Copies one or more files to a new destination

The COPY command copies one or more files to a new disk drive, directory, or filename.

Format

```
COPY source_file [/V][/A][/B]
target_file [/V][/A][/B]

COPY source1+source2[/V][/A][/B][...]
target_file [/V][/A][/B]
```

source_file Specifies the file to be copied.

target_file The name of the destination file.

/V Uses disk verification to check whether a successful copy occurred.

/A Informs COPY that the preceding file was an ASCII file.

/B Informs COPY that the preceding file was a binary file.

source1+source2 Indicates that you can use any number of source files.

Notes

The COPY command fully supports DOS wildcard characters. To combine multiple files into one file, use the plus sign (+) between the desired source files.

COPY will not allow you to copy a file over itself. The names of source and target files must differ in some way. If the target file already exists, the COPY command will overwrite the file.

The COPY command only copies files contained in the current directory or a specified DOS directory. To copy files that reside in several subdirectories, use XCOPY.

The /V parameter adds processing overhead but prevents a hardware error from rendering inconsistent the contents of the source and target files.

Examples

The following command copies a spreadsheet file called BUDGET.DAT from the Lotus 1-2-3 directory to a floppy disk in drive A:

```
C:\> COPY \123\BUDGET.DAT A: <ENTER>
```

The following command copies the contents of the CONFIG.SYS file to a file with the same name in drive B:

```
C:\> COPY A:CONFIG.SYS B:CONFIG.SYS <ENTER>
```

This command is identical in function to the following:

```
C:\> COPY A:CONFIG.SYS B: <ENTER>
```

or to the commands:

```
C:\> B: <ENTER>
B:\> COPY A:CONFIG.SYS <ENTER>
```

The following command copies all the files on drive A to drive B:

```
C:\> COPY A:*.* B:*.* <ENTER>
```

The following command uses the plus sign to append files TWO.DAT and THREE.DAT to file ONE.DAT, thereby creating a file called FOUR.DAT:

```
C:\> COPY ONE.DAT+TWO.DAT+THREE.DAT FOUR.DAT <ENTER>
```

Messages

Cannot do binary reads from a device

You have placed the /B parameter after a device name. When COPY performs a binary copy, the command does not use the CTRL-Z (^Z) character to indicate an end-of-file. Since CTRL-Z is the only way to tell COPY that you have entered your required information from a device, COPY cannot perform a binary copy from a device.

Contents of destination lost before copy

The name of the destination file is also specified as a source file in the COPY operation. COPY became confused and has overwritten the file's contents.

File cannot be copied onto itself

The source and destination for a file-copy operation are the same. In most cases, this error occurs when you have omitted a disk-drive specifier or subdirectory in the target file's name.

CTTY

Modifies standard input (stdin) to point to alternate device

The CTTY command changes standard I /O from a default device to an alternate device. Valid device names include AUX, COM1, and COM2. To return standard input to the console device, the command CTTY CON must be issued through the auxiliary device.

Format

```
CTTY device_name
```

device_name The name of the device you want for standard input.

Notes

Most end users have no need to issue this command. If you invoke CTTY without your computer attached to the serial port, you will have to reboot DOS in order to continue.

Example

This command sets the standard input/output to COM1:

```
A> CTTY COM1:
```

Message

Invalid device

The command line specifies either an invalid device name or no device name. Invoke CTTY only when your system has a terminal device attached to a serial port.

DATE

Sets the system date

The DATE command sets the system date DOS uses to assign date stamps to each file you create or modify.

Format

```
DATE [mm-dd-yy]
DATE [dd-mm-yy]
DATE [yy-mm-dd]
```

mm The desired month (1 to 12).

dd The desired day (1 to 31).

yy The desired year (80 to 99). DATE also allows you to include the century, in the form 19yy or 20yy.

Notes

The mm-dd-yy date format depends on the COUNTRY specifier in CONFIG.SYS (see CONFIG). You can also use slashes (/) instead of dashes (-). If you do not specify a date, DATE displays the current date.

Prior to DOS 3.3, the DATE command did not modify the AT computer's system clock. Users of DOS versions earlier than 3.3 must use the Setup disk provided with the *Guide to Operations* in order to change the AT system clock.

The actual date is an optional command-line parameter. If you omit the date, DATE prompts you for it. Although you do not have to type in a date, doing so is good practice. DOS uses the system's current date and time to time-stamp files whenever a file is created or modified.

Examples

Since the command line in the following example does not include the date, DATE prompts the user for it:

```
C:\> DATE <ENTER>
The current date is: Mon 04-05-1993
Enter the new date: (mm-dd-yy)
```

If you simply want to display the system date without modifying it, press the ENTER key at the date prompt. DATE leaves the system date unchanged.

In the following example, DATE sets the system date to December 8, 1993:

```
C:\> DATE 12/08/93 <ENTER>
```

This command has the same effect as

```
C:\> DATE 12/08/1993 <ENTER>
```

Message

Invalid date
Enter new date (mm-dd-yy):

Make sure that you enter the current system date in the correct format, with the month preceding the day. Simply retype the command and press ENTER.

DBLSPACE

Compresses information stored on a disk, increasing the disk's storage capacity

DBLSPACE compresses the information stored on a disk, essentially doubling the disk's storage capacity. DBLSPACE works with hard or floppy disks. After you compress a disk, you work with the disk's files, just as you always have. In other words, DOS transparently compresses and decompresses files as you use them, so your operations are unaffected.

Format

```
DBLSPACE [/CHKDSK [/F] [Drive:]]
  [/COMPRESS Drive: [/NEWDRIVE=Drive:] [/RESERVE=Megabytes]]
  [/CONVSTAC=StackerVolume: [/NEWDRIVE=Drive:] [/CVF=xxx]]
  [/CREATE Drive: [/NEWDRIVE=Drive:] /SIZE=Megabytes]
          [RESERVE=Megabytes]]
  [/DEFRAGMENT [Drive:]]
  [/DELETE Drive:]
  [/FORMAT Drive:]
  [/INFO Drive:]
  [/LIST]
  [/MOUNT[=xxx] Drive: /NEWDRIVE=Drive:]
  [/UNMOUNT Drive:]
  [/RATIO[=x.x] [Drive: | /ALL]]
  [/SIZE[=Megabytes | /RESERVE=Megabytes] Drive:]
```

/CHKDSK Directs DBLSPACE to examine the internal structure of the compressed volume file.

/COMPRESS Directs DBLSPACE to compress the files on the drive specified. DBLSPACE uses the drive letter specified after the /NEWDRIVE switch as the uncompressed drive volume. The /RESERVE switch specifies the amount of disk space in megabytes that DBLSPACE must leave on the uncompressed volume.

/CONVSTAC Directs DBLSPACE to convert a Stacker disk volume into DBLSPACE format. The /NEWDRIVE switch specifies the drive letter of the newly compressed drive. The /CVF switch specifies the three-digit (000 through 254) file extension of the compressed volume file.

/CREATE Directs DBLSPACE to create a compressed drive using free space on the specified drive.

/DEFRAGMENT Directs DBLSPACE to defragment the files stored on the compressed volume specified.

/DELETE Directs DBLSPACE to delete a compressed drive.

/FORMAT Directs DBLSPACE to format a compressed drive.

/INFO Directs DBLSPACE to display specifics about a compressed drive.

/LIST Directs DBLSPACE to display information about a system's compressed and uncompressed drives.

/MOUNT Directs DBLSPACE to mount a compressed drive, preparing the drive for use. The xxx values correspond to the three-digit file extension of the drive to mount. The Drive: parameter specifies the disk containing the compressed volume file. The /NEWDRIVE switch lets you specify the desired drive letter for the newly mounted drive.

/UNMOUNT Directs DBLSPACE to dismount a compressed drive.

/RATIO Directs DBLSPACE to change a drive's compression ratio. The x.x parameter specifies the desired compression ratio from 1.0 to 16.0. The Drive: parameter specifies the disk drive for which you want to change the compression ratio.

/SIZE Directs DBLSPACE to change the size of a compressed drive.

Notes

The first time that you invoke the DBLSPACE command, DBLSPACE executes the DBLSPACE setup and installation, typically compressing your hard disk. After that, most users will not invoke DBLSPACE again. However, as you can see, DBLSPACE contains several command-line switches that power users may include to further customize their disk compression.

DBLSPACE requires DOS 6.

Examples

If you have not yet invoked DBLSPACE to compress your disk, the following DBLSPACE command will begin the software installation and disk compression:

```
C:\> DBLSPACE <ENTER>
```

Simply perform the operations as prompted. Depending on your disk's capacity and current files, the compression operation may require an hour or more.

The following DBLSPACE command uses the /INFO switch to direct DBLSPACE to provide more specifics about a compressed drive:

```
C:\> DBLSPACE /INFO <ENTER>
DoubleSpace is examining drive C.
Compressed drive C (DOS 6 DISK) was created on 04-24-1993 at
12:08am. Drive C
is stored on uncompressed drive H in the file DBLSPACE.000.

                   Compressed    Uncompressed
                     Drive C        Drive H

   Total space:    337.11 MB      190.79 MB
   Space used:     221.69 MB      187.66 MB
   Space free:     115.42 MB**      3.13 MB

   The actual compression ratio is 1.7 to 1.

** based on estimated compression ratio of 2.0 to 1.
```

Likewise, this command directs DBLSPACE to display information about a system's available drives:

```
C:\> DBLSPACE /LIST <ENTER>
Drive Type                   Total Free  Total Size  CVF Filename
-------------------------  ----------  ----------  -------------
A     Floppy drive              0.00 MB     0.69 MB
C     Compressed hard drive 115.42 MB   337.11 MB  H:\DBLSPACE.000
E     DoubleSpace
F     DoubleSpace
G     DoubleSpace
H     Local hard drive          3.14 MB   190.79 MB
```

The following command directs DBLSPACE to check the internal consistency of the compressed drive C using the /CHKDSK switch:

```
C:\> DBLSPACE /CHKDSK C: <ENTER>
```

If DBLSPACE encounters errors during the CHKDSK operation, repeat the command including the /F switch.

The following DBLSPACE command uses the /DEFRAGMENT switch to defragment the compressed drive D:

```
C:\> DBLSPACE /DEFRAGMENT D: <ENTER>
```

DEFRAG

Eliminates file fragmentation to improve performance

As you create, delete, and change files, a file's contents may get stored in locations spread out across your disk. Files whose contents are not stored in consecutive locations are fragmented files. Such files decrease system performance by increasing the number of disk operations required to read the file. The DEFRAG command moves files on your disk to place them in consecutive storage locations.

Format

```
DEFRAG [Drive:][/B][/F][/S[:]SortOrder]
[/SKIPHIGH][/V]
```

or

```
DEFRAG [Drive:][/B][/SKIPHIGH][/U][/V]
```

Drive: Specifies the drive letter corresponding to the disk you want to defragment. If you don't specify a drive letter, DEFRAG will prompt you for one.

/B Directs DEFRAG to boot your system after it completes. If you use FASTOPEN, include the /B switch when you invoke DEFRAG.

/F Directs DEFRAG not only to defragment files but also to remove any holes (unused clusters) between files.

/S[:]SortOrder Specifies the order in which DEFRAG sorts files in each directory:

N	Name A to Z
E	Extension A to Z
D	Date oldest first
S	Size smallest first
–N	Name Z to A
–E	Extension Z to A
–D	Date newest first
–S	Size largest first

If you don't specify a sort order, DEFRAG leaves the directory order unchanged.

/SKIPHIGH Directs DEFRAG to load into conventional memory. By default, DEFRAG loads itself into upper memory if the memory space is available.

/U Directs DEFRAG to defragment files and leave existing holes (unused clusters) that exist between files.

/V Directs DEFRAG to verify the data it writes to disk during a cluster move operation is recorded correctly. Using /V slows down the defragmentation operation.

Notes

If your programs or files seem to take longer to load, or your disk's activation light seems to remain on for longer periods of time, your files may be fragmented. As a general rule, if you regularly create, delete, and change files, you should defragment your disk once a month.

To support batch processing, DEFRAG uses the following exit status values:

Exit Status	Meaning
0	Successful defragmentation
1	Internal error
2	Disk has 0 free clusters; DEFRAG requires at least one
3	User CTRL-C
4	General error
5	Disk read error
6	Disk write error

Exit Status	Meaning
7	Cluster error; use CHKDSK /F
8	Memory error
9	Insufficient memory

DEFRAG requires DOS 6

Examples

DEFRAG is a menu-driven utility program. As such, most users simply invoke DEFRAG as follows, using its menus to select desired options:

```
C:\> DEFRAG <ENTER>
```

If you are using DOS FASTOPEN, invoke DEFRAG using the /B switch:

```
C:\> DEFRAG /B <ENTER>
```

DEL

Deletes one or more files from disk

Use the DEL command to delete from the disk a file that no longer is needed.

Format

```
DEL [drive:][path]file_name[.ext][/P]
```

drive: Specifies the disk drive that contains the file to be deleted. If you omit this parameter, DEL uses the default drive.

path Specifies the name of the subdirectory that contains the file to be deleted. If you omit the path, DEL uses the current directory.

file_name[.ext] The name of the file to be deleted. DEL fully supports DOS wildcard characters.

/P A DOS 4 or later switch that directs DEL to prompt you before deleting the file specified.

Notes

For compatibility with earlier versions of DOS, you can invoke the DEL command as ERASE.

Unless you explicitly use a drive or path specifier, DEL deletes files in the current directory only. You can delete several files simultaneously by including each filename on the same command line.

You cannot use DEL to remove subdirectories: use RMDIR.

If you attempt to delete all the files in a directory, DOS asks

```
Are you sure (Y/N)?
```

to ensure that you truly want the command performed. To proceed, press **Y** and ENTER; otherwise, press **N** and ENTER.

Using DOS 4 or later, if you invoke DEL with wildcard characters and the /P parameter, DOS prompts you for each individual file to determine whether you want to delete it:

```
FILENAME.EXT, Delete (Y/N)?
```

Type **Y** and press ENTER to delete the file. If you type **N** and press ENTER, DEL will leave the file on disk.

If you are using DOS 5 or later, you may be able to use the UNDELETE command to restore one or more recently deleted files.

Examples

In the following example, DEL erases the contents of the CONFIG.OLD file from drive B:

```
C:\> DEL B:CONFIG.OLD <ENTER>
```

Similarly, the following command:

```
C:\> DEL AUTOEXEC.OLD <ENTER>
```

deletes a file from within the current subdirectory.

If you are using DOS 4 or later, you can use the /P parameter to tell DEL to prompt you before deleting a file:

```
C:\> DEL *.BAK /P <ENTER>
```

Messages

Access Denied

Either you have tried to delete a file marked as read-only by ATTRIB +R, or DOS has located the file in a directory specified in the data-file search path (see APPEND).

Are you sure (Y/N)?

You have issued DEL with the *.* wildcard characters. Since this command can have a devastating impact on your system if invoked in error, DEL prompts you to verify your intentions.

DELTREE

Deletes all of the files and subdirectories in a directory

The DELTREE command lets you remove a directory, its files, and lower-level subdirectories in one step.

Format

```
DELTREE [/Y] Pathname
```

/Y Directs DELTREE not to prompt you before deleting the directory.

Pathname Specifies the directory DELTREE is to delete.

Notes

DELTREE will delete all of the files in a directory and its subdirectories, regardless of the file's current attribute settings. If DELTREE successfully removes a directory, DELTREE returns the exit status value 0.

NOTE

DELTREE supports the DOS wildcard characters. However, use wildcards with caution. The wrong wildcard combination could quickly erase many directories from your disk.

Examples

The following DELTREE command deletes the directory OLDNEWS as well as the subdirectories OLDNEWS contains:

```
C:\> DELTREE OLDNEWS <ENTER>
```

If you do not include the /Y switch, DELTREE will prompt you to type **Y** to delete a directory or **N** to leave the directory alone, as shown here:

```
C:\> DELTREE /Y OLDNEWS <ENTER>
Delete directory "OLDNEWS" and all its subdirectories? [yn]
```

If you type **Y**, DELTREE will delete the directory. If you type **N**, DELTREE will leave the directory unchanged. To eliminate the prompts, invoke the DELTREE command with the /Y switch.

Message

Delete directory "*PATHNAME*" and all its subdirectories? [yn]

You have invoked DELTREE without the /Y switch and DELTREE is asking you to verify that you want to delete the specified directory. If you type **Y**, DELTREE will delete the directory and subdirectories. If you type **N**, DELTREE will leave the directory alone.

DIR

Displays a directory listing of files

A directory is a list of filenames. DIR displays the names of all files in the directory specified.

Format

```
DIR [file_spec][/P][/W]
```

or, using DOS 5 or later:

```
DIR [file_spec][/P][/W][A:attributes]
  [/O:sort_order][/S][/B][/L]
```

file_spec The file(s) for which DIR is to display a directory listing. The file specification can include a disk drive identifier and path name. If you do not place a file specification in the command line, DIR displays a listing of all files in the current directory. DIR fully supports DOS wildcard characters.

/P Pauses after each screenful of information.

/W Displays the files in short form (filename only), with five filenames across the screen.

/A:attributes A DOS 5 or later switch that tells DIR to display only files that meet specific attribute criteria.

/O:sort_order A DOS 5 or later switch that tells DIR to display the directory listing in an order other than by name.

/S A DOS 5 or later switch that displays the files that reside in lower-level subdirectories.

/B A DOS 5 or later switch that displays filenames and extensions only.

/L A DOS 5 or later switch that displays filenames in lowercase letters.

Notes

By default, the DIR command displays each file's complete name, file size in bytes, and date and time of creation (or last modification). The DIR command also displays the amount of free disk space in bytes, and in DOS 5 also displays the total number of bytes in files in the directory. By default, DIR does not display hidden files.

DIR always displays the drive letter and directory name in which the files are stored. When you use the /B parameter available in DOS 5 or later, the drive letter and directory name are not displayed.

If you simply invoke DIR with a filename, like this:

```
C:\> DIR FILENAME <ENTER>
```

the extension defaults to *.

When using DOS 5 or later, you can use the /O:sort_order parameter to modify the order in which files are displayed. Valid sort orders include the following:

N	A to Z by name
–N	Z to A by name
C	By compression ratio smallest to largest (DOS 6)
–C	By compression ratio largest to smallest (DOS 6)
E	A to Z by extension
–E	Z to A by extension
D	By date oldest to newest
–D	By date newest to oldest
S	By size smallest to largest
–S	By size largest to smallest
G	Directories grouped before files
–G	Directories grouped after files

If you are using DOS 5 or later, you can also use the environment entry DIRCMD to define the default format DIR uses to display files. For example, the following SET command assigns DIRCMD the parameters /L and /S (for lowercase and subdirectories):

```
C:\> SET DIRCMD=/L/S <ENTER>
```

With DOS 5 or later, you can use the /A:attributes parameter to control which files DIR displays. Only the names of those files whose file attributes match those in the attributes string are listed. Valid attribute values include the following:

H	Hidden files
–H	Files that are not hidden files
S	System files
–S	Files that are not system files
D	Directories
–D	Files that are not directories
A	Files needing archiving
–A	Files already archived
R	Read-only files
–R	Files that are not read-only

You can combine attributes as well. For example, /A:A–R tells DIR to display all files requiring archiving that are not read-only files.

Examples

The following example causes the DIR command to display a directory listing of all files on drive A. The two lines are functionally equivalent and produce the same results:

```
C:\> DIR A: <ENTER>

C:\> DIR A:*.* <ENTER>
```

If many files exist on drive A, they may scroll off the screen during the directory listing. If this happens, invoke DIR as follows:

```
C:\> DIR A:*.* /P <ENTER>
```

Whenever DIR completes a screenful of files, it will pause, displaying a prompt similar to the following:

```
Strike a key when ready . . .
```

When this occurs, press any key for DIR to continue.

The file specification of the DIR command can be as specific as you would like. The following command displays the directory listing of the CONFIG.OLD file, which resides in the subdirectory on the current disk:

```
C:\> DIR \DOS\CONFIG.OLD <ENTER>
```

If you are using DOS 5 or later, use the following command to tell DIR to display a directory listing sorted by size:

```
C:\> DIR /O:S <ENTER>
```

The following DOS 5 or later command tells DIR to display all files that reside on your disk, showing their names in the lower case:

```
C:\> DIR \*.* /S /L <ENTER>
```

Finally, this DOS 5 or later command tells DIR to display only the names of subdirectories that reside in the current directory:

```
C:\> DIR /A:D <ENTER>
```

Message

Strike a key when ready . . .

You have invoked DIR with the /P parameter, which suspends display of the directory listing with each screenful of information. Simply press a key to continue the directory listing, or press CTRL-BREAK to terminate the command.

DISKCOMP

Compares two floppy disks

The DISKCOMP command compares the contents of two floppy disks and displays differences if the disks are not identical.

Format

```
DISKCOMP[primary_drive:  [secondary_drive]][/1][/8]
```

primary_drive Specifies the first drive to be used for the disk comparison. If you do not specify a primary drive, DISKCOMP uses the default.

secondary_drive Specifies the second drive to be used for the disk comparison. If you do not specify a secondary drive, DISKCOMP uses the default.

/1 Tells DISKCOMP to perform a single-sided disk comparison.

/8 Tells DISKCOMP to perform an eight-sector-per-track disk comparison.

Notes

If you have a single-floppy system, the DISKCOMP command will perform a single-drive comparison, prompting you to enter the source disk and target disk at the correct time.

If the contents of the disks are identical, DISKCOMP displays the following message:

```
Compare OK
```

Otherwise, DISKCOMP displays in hexadecimal values the location (disk's side and track) of the differences.

Many users issue the DISKCOMP command immediately after using DISKCOPY to copy the contents of one disk to another. DISKCOMP detects any errors that occurred during the disk copy.

DISKCOMP returns the following exit status values:

Exit Status	Meaning
0	Disks compare OK
1	Disks are not the same
2	User termination via CTRL-C
3	Unrecoverable disk error
4	Invalid syntax, insufficient memory, or invalid drive

Examples

In the following example, DISKCOMP compares the contents of the disk in drive A to those of drive B:

```
C:\> DISKCOMP A: B: <ENTER>
```

If there are errors, DISKCOMP displays the locations of the differences:

```
Compare error on side n track n
```

If you need to compare single-sided disks, use the following command:

```
C:\> DISKCOMP A: B: /1 <ENTER>
```

Messages

Cannot DISKCOMP to or from a Network drive

One of the disk drives you specified in the DISKCOMP command line references a network drive. DOS will not allow you to perform DISKCOMP operations to a network disk drive.

Compare another diskette (Y/N)?

DISKCOMP has completed one disk-comparison operation and is asking whether you want to perform another. If you do, type **Y**; DISKCOMP will prompt you to enter the disks to compare. To terminate the DISKCOMP command, simply type **N** and then press ENTER.

Compare error on side *n*, track *n*

DISKCOMP has located differences between the two disks at the location specified.

Compare OK

The two disks are identical.

Comparing *n* tracks
***n* sectors per track, *n* side(s)**

DISKCOMP is telling you the format of the disks it is comparing. DISKCOMP compares two disks sector by sector, track by track, side by side. This message simply tells you the number of tracks, sides, and sectors that DISKCOMP will examine on each disk.

Drive *n* not ready

The disk drive latch on the drive specified is probably open. Make sure that a floppy disk has been inserted into the drive and that the door is closed.

First diskette bad or incompatible

The disk in the first drive specified in the DISKCOMP command line is either damaged or incompatible with the disk drive. For example, you cannot examine a 1.2Mb disk in a 360Kb floppy disk drive.

Insert FIRST diskette in drive *n*:
Press any key when ready ...

DISKCOMP is prompting you to insert one of the two disks to be compared into the drive specified. If you are performing a single disk drive comparison, place the first disk in the drive specified.

Insert SECOND diskette in drive *n*:
Press any key when ready ...

DISKCOMP is prompting you to insert one of the two disks into the drive specified. If you are performing a single disk drive comparison, put a different floppy disk within the drive specified.

Invalid drive specification
Specified disk does not exist
or is non-removable

The disk drive you specified is either invalid or a fixed disk. DISKCOMP does not allow you to compare fixed disks. If the drive specified is not a fixed disk, make sure that you are using a valid disk drive letter followed by a colon.

Invalid parameter
Do not specify filename(s)
Command format: DISKCOMP d: d:[/1][/8]

The DISKCOMP command line is invalid. Examine the DISKCOMP command format shown in the "Format" section and reissue the command.

Second diskette bad or incompatible

The disk in the second drive specified in the DISKCOMP command line is either damaged or incompatible with the disk drive. For example, you cannot examine a 1.2Mb disk in a 360Kb floppy disk drive.

DISKCOPY

Copies a source floppy disk to a target disk

The DISKCOPY command copies the contents of one floppy disk to another, creating an identical copy.

Format

```
DISKCOPY[source_drive:[target_drive]][/1][/V]
```

source_drive Specifies the disk drive that contains the floppy disk to be copied. If you do not specify a source drive, DOS uses the default drive.

target_drive Specifies the disk drive that contains the disk to be copied to. If you do not specify a target drive, DOS uses the default drive.

/1 Tells DISKCOPY to copy only the first side of the source disk to the target disk.

/V Enables disk verification (DOS 5 or later). Disk verification double-checks that all information written to the disk is recorded correctly.

Notes

If you have a single-floppy system, DISKCOPY performs a single-drive copy, prompting you to enter the source and target disks at the correct time.

DISKCOPY supports the following exit status values:

Exit Status	Meaning
0	Successful disk copy
1	Non-fatal disk read/write error
2	Disk copy ended by user CTRL-C
3	Disk copy ended due to fatal error
4	Initialization error prevented copy

DISKCOPY destroys the previous contents of the target disk. If the source and target disks are different types of disks (a 360Kb and a 1.2Mb disk, for example), DISKCOPY displays an error message and terminates.

If the target disk has not been formatted, DISKCOPY formats it during the copy operation.

Do not use DISKCOPY with joined or substituted disks (see JOIN and SUBST).

Examples

The following command assumes that you have two compatible floppy disk drives on your system:

```
C:\> DISKCOPY A: B: <ENTER>
```

After DISKCOPY begins copying a disk, it reads several tracks of data from the source and then writes them to the target disk. In a single-floppy drive system, DISKCOPY repeats this process, prompting you for the source and target disks, as shown here:

```
A:\> DISKCOPY <ENTER>
Insert SOURCE diskette in drive A:
Press any key when ready
Copying 40 tracks
9 Sectors/Track, 2 Side(s)

Insert TARGET diskette in drive A:
Press any key when ready

Insert SOURCE diskette in drive A:
Press any key when ready
```

```
Insert TARGET diskette in drive A:
Press any key when ready

Copy another diskette (Y/N)?
```

Messages

Cannot DISKCOPY to or from a Network drive

One of the disk drives specified in the DISKCOPY command line references a network disk drive. DOS does not allow you to use a network device in a DISKCOPY operation.

Copying *n* tracks
n sectors per track, *n* side(s)

DISKCOPY is telling you the format of the disk it is duplicating. This message is purely informational.

Copy process ended

The DISKCOPY operation has failed. Normally, DISKCOPY precedes this error message with a message describing the reason for the failure.

Drive types or diskette types not compatible

Either the two disks or the disk drives specified in the DISKCOPY command are incompatible.

Formatting while copying

The target disk in the DISKCOPY operation was unformatted. DISKCOPY must format the disk before it can be used by DOS. In this case, DISKCOPY is simply informing you that it is formatting the disk as it copies.

Insert source diskette in drive *n*:
Press any key when ready...

DISKCOPY is prompting you to insert the disk you want to copy (the source disk) into the drive specified. After doing so, press any key to continue.

Insert target diskette in drive *n*:
Press any key when ready...

DISKCOPY is prompting you to insert the disk you want to copy to (the target disk) into the drive specified. After doing so, press any key to continue.

Invalid parameter
Do not specify filename(s)
Command Format: DISKCOPY D: D:[/1]

The DISKCOPY command line contains invalid entries. Examine the format of the DISKCOPY command shown here and reissue the correct DISKCOPY command.

Source diskette bad or incompatible

The source disk is damaged or incompatible with the disk drive in which it resides. Make sure that you are not trying to use a 1.2Mb disk in a 360Kb disk drive. Reissue the DISKCOPY command; if it fails again, your source disk may be severely damaged.

Target diskette bad or incompatible

The target disk is damaged or incompatible with the disk drive in which it resides. Make sure that you are not trying to use a 1.2Mb disk in a 360Kb disk drive. Reissue the DISKCOPY command; if it fails again, your target disk may be severely damaged.

Target diskette is write protected

The target disk has a write-protect tab preventing DISKCOPY from writing to it.

Target diskette may be unusable

DISKCOPY was unable to create an identical copy of the source disk. This error normally occurs when the target disk contains damaged locations that cannot be used. To ensure a complete disk copy, repeat the DISKCOPY command with a different target disk.

Unrecoverable read error on drive *n*:
side *n*, track *n*

DISKCOPY was unable to read the track of the source disk specified. Since DISKCOPY copies one disk to another, sector by sector, track by track, the missing information will make your disk copy incomplete. In most cases, this error is associated with copy-protected software or a damaged source disk.

Unrecoverable write error on drive *n*:
side *n*, track *n*

DISKCOPY was unable to write the track of the target disk specified. Since DISKCOPY copies one disk to another sector by sector, track by track, the missing information will make your disk copy incomplete. In most cases, this error is associated with a damaged target disk.

DOSKEY

Recalls previously entered commands

DOSKEY is a DOS 5 or later command that lets you recall and edit previously entered commands and create macros (similar to batch files) that DOS stores in memory.

Format

```
DOSKEY[/REINSTALL][/BUFSIZE=size][/MACROS]
   [/HISTORY][/INSERT | /OVERSTRIKE]
   [macro_name= macro]
```

/REINSTALL Installs a new copy of the DOSKEY program in memory.

/BUFSIZE=size Specifies the size of the command and macro buffer in memory. If you are creating many macros, use a buffer size of 4096. The default size is 512.

/MACROS Displays the current list of macros.

/HISTORY Displays a list of all commands stored in the buffer (the command history).

/INSERT Lets you remain in insert mode after you press ENTER.

/OVERSTRIKE Disables insert mode.

macro_name=macro Lets you create a macro.

Notes

With DOSKEY enabled, you can use the keyboard arrow keys to toggle through previously entered commands. When the command you want is displayed, execute it by pressing ENTER. To edit the command, use the following:

Key	Action
UP ARROW	Recalls the command preceding the one displayed
DOWN ARROW	Recalls the command after the one displayed
PAGE UP	Recalls the oldest command in buffer
PAGE DOWN	Recalls last command placed in buffer
LEFT ARROW	Moves cursor left one character
RIGHT ARROW	Moves cursor right one character
CTRL-LEFT ARROW	Moves cursor back one word
CTRL-RIGHT ARROW	Moves cursor forward one word
HOME	Moves cursor to start of line
END	Moves cursor to end of line
ESC	Clears command from display
F7	Displays all commands in memory, preceded by a line number
ALT-F7	Erases all commands stored in memory
F8	Searches memory for command matching partial command line displayed
F9	Specifies a command line by number
F10	Displays all macros stored in memory
ALT-F10	Clears all macros

Macros are similar in concept to DOS batch files. A macro can contain one or more commands. When you create a macro, it must fit on one page. DOS stores the macro in RAM. To execute a macro, enter its name.

Like DOS batch files, macros support command-line parameters. Instead of using the symbols %0 to %9, macros use $1 through $9. Macros also support the following metacharacters:

Character	Meaning
$G or $g	DOS output redirection operator
$L or $l	DOS input redirection operator
$B or $b	DOS pipe operator

Character	Meaning
$T or $t	Command separator
$*	Entire command line, minus the macro name

Examples

The following command loads DOSKEY with a 4096-byte buffer for commands and macros:

```
C:\> DOSKEY /BUFSIZE=4096 <ENTER>
```

This command

```
C:\> DOSKEY /HISTORY <ENTER>
```

tells DOSKEY to display all of the commands in the command buffer.

Likewise, the following command tells DOSKEY to display all of its macros:

```
C:\> DOSKEY /MACROS <ENTER>
```

To create a macro named CP that abbreviates the COPY command, use the following:

```
C:\> DOSKEY CP=COPY $1 $2 <ENTER>
```

As you can see, the CP macro uses the symbols $1 and $2 to access its command-line parameters. Once you create the macro, you access it by using the name CP.

Finally, the following command

```
C:\> DOSKEY HIST=DOSKEY /HISTORY $T DOSKEY /MACROS <ENTER>
```

creates a macro (HIST) that displays not only the commands currently stored in the DOSKEY buffer but also the current macros.

HIST uses the $T metacharacter to separate the two commands.

Messages

Can't change BUFSIZE

Your DOSKEY command included the /BUFSIZE parameter after DOSKEY was installed. Once DOSKEY is installed, you cannot change the buffer size. To change the buffer size, you must restart DOS.

DOSKEY installed

The DOSKEY installation was successful.

Invalid macro definition

The DOSKEY command line does not contain a valid macro. The format for macros is as follows:

```
macro_name=macro_text
```

DOSSHELL

Starts the DOS shell user interface

Starting with version 4, DOS provides a menu-driven shell interface that gives you a graphically oriented, user-friendly alternative to the DOS command line for giving commands. The shell was changed in appearance and style of function starting in DOS 5.

Format

```
DOSSHELL [/T[:L|M|H {Resolution]]] [/B]
         [/G{:L|M|H [Resolution]]]
```

/T Starts the DOS shell in low-, medium-, or high-resolution text mode.

/B Starts the DOS shell in black and white.

/G Starts the DOS shell in low-, medium-, or high-resolution graphics mode.

Notes

The file DOSSHELL.INI contains the shell's program group, screen color, resolution, and other settings. If you move the file from the DOS directory, place a SET command in your AUTOEXEC.BAT file that defines the file's new directory, as shown here:

```
SET DOSSHELL=C:\NEWDIR
```

Example

Most users invoke the DOSSHELL from the command prompt as follows and then use the shell's Options menu to specify graphics or text mode settings:

```
C:\> DOSSHELL <ENTER>
```

ECHO

Displays or suppresses batch command messages

By default, whenever you execute DOS batch files, DOS displays the name of each command as it executes. You can stop this command-name display by using the ECHO OFF command. In addition, many batch files use ECHO to display messages to the user.

Format

```
ECHO[ON | OFF | message]
```

ON Enables the display of batch commands as they execute.

OFF Disables the display of batch commands as they execute.

message Contains the text ECHO is to display.

Notes

If you are using DOS 3.3 or later, you can suppress command-name display by preceding command names in a batch file with an @ sign, as shown here:

```
@ECHO OFF
DATE
```

Many batch files use ECHO to display messages for users. For example, the following batch file displays the message "Hello there!"

```
@ECHO OFF
ECHO Hello there!
```

If you aren't using DOS 5 and you want ECHO to display a blank line, you must place the ALT-255 keyboard combination after the ECHO command, as follows:

```
@ECHO OFF
ECHO Skipping one 1
ECHO ALT-255
ECHO Done
```

To enter the ALT-255 keyboard combination, hold down the ALT key. At the same time, *using the numeric keypad*, type in the value **255**. When you release the ALT key, you will see the cursor move one position to the right. The ALT-255 keyboard combination creates a special blank character. When this character appears in an ECHO command line, ECHO displays a blank line.

Beginning with DOS 3.3, you can direct ECHO to display a blank line by simply placing one of several special characters immediately after the ECHO command. You can use a period, or the characters +, /,[,], or a colon. The following example uses a period:

```
@ECHO OFF
ECHO Skipping one line
ECHO.
ECHO Done
```

Examples

The following batch file uses ECHO to display several messages on the screen:

```
@ECHO OFF
ECHO *********************
ECHO * Batch File Example *
ECHO *                    *
ECHO * Messages displayed *
ECHO *     by  ECHO       *
ECHO *********************
```

When you invoke this batch file, it displays the following:

```
*********************
* Batch File Example *
*                    *
* Messages displayed *
*     by  ECHO       *
*********************
```

EDIT

Invokes the DOS full-screen editor

Beginning with DOS 5, DOS provides a full-screen editor that lets you quickly create or change ASCII files, while viewing the file's contents on your screen.

Format

```
EDIT [Filename] [/B] [/G] [/H] [/NOHI]
```

Filename Specifies the name of the file you want to edit.

/B Directs EDIT to display its output in black and white.

/G Directs EDIT to perform fast CGA screen updates.

/H Directs EDIT to display its output using the maximum number of screen lines that your video card supports.

/NOHI Directs EDIT to not use high intensity display attributes.

Notes

To use EDIT, the file QBASIC.EXE must reside in the current directory or within a directory specified in the command path.

Within EDIT, you can use menu options to save and print the current file's contents.

Example

The following command invokes EDIT with the file CONFIG.SYS:

```
C:\> EDIT \CONFIG.SYS <ENTER>
```

EXIT

Terminates a secondary command processor

Many application programs provide an option that lets you temporarily exit the application to the DOS prompt. After you finish issuing DOS commands, you can return to the application by typing **EXIT**.

Format

```
EXIT
```

Notes

The DOS shell lets you temporarily exit the shell to the DOS prompt. When you are ready to resume work in the shell, type EXIT at the prompt and press ENTER.

EXPAND

Expands a DOS compressed file

To reduce the number of distribution disks, many DOS 5 files are shipped on disk in a compressed format. If you examine your distribution disks, you will see files whose last letter of their extension is an underscore (_) character, such as UNDELETE.EX_. During the DOS 5 installation, SETUP expands these files onto your hard disk as necessary. Should you ever need to copy one of these files to your disk after installation, you will need to expand it.

Format

```
EXPAND source_file target_file
```

source_file The complete path name of the file whose contents you want to expand.

target_file The name of the expanded file.

Notes

You cannot use a compressed file until you expand it.

Example

The following expands the file UNDELTE.EX_ on the floppy disk in drive A to your DOS directory as UNDELETE.EXE.

```
C: EXPAND A:UNDELETE.EX_ \DOS\UNDELETE.EXE <ENTER>
```

FASTHELP

Provides online help for DOS commands

In versions 5 and 6 DOS provides onscreen help screens explaining its commands.

Format

`FASTHELP [CommandName]` (in DOS 6)

or

`HELP [CommandName]` (in DOS 5)

CommandName Specifies a DOS command for which you want to display help text.

Notes

In DOS 6, the command is FASTHELP. In DOS 5, it is called HELP. Both DOS 5 and DOS 6 let you obtain help on a specific command by invoking the command followed by the /? switch, as shown here:

```
C:\> CLS /? <ENTER>
Clears the screen.


CLS
```

If you do not include a command name, FASTHELP displays a single-line description of each command. In DOS 5, see also the HELP command.

Examples

The following command displays help text on CHKDSK:

```
C:\> FASTHELP CHKDSK <ENTER>
Checks a disk and displays a status report.

CHKDSK [drive:][[path]filename] [/F] [/V]

  [drive:][path] Specifies the drive and directory to check.
  filename       Specifies the file(s) to check for fragmentation.
  /F             Fixes errors on the disk.
  /V             Displays the full path and name of every file on the disk.

Type CHKDSK without parameters to check the current disk.
```

In DOS 5, the following command displays help text on CHKDSK:

```
C:\> HELP CHKDSK <ENTER>
```

Finally, the following command displays a one-line summary of each DOS command:

```
C:\> FASTHELP <ENTER>
```

Messages

Help is not available for this command

You have probably misspelled the command for which you want help.

Help file cannot be found

DOS could not find the file FASTHELP.HLP. Make sure the file resides in the DOS directory or a directory specified in the command path.

FASTOPEN

Increases directory search performance

Whenever you invoke a program or open a file, DOS must search the disk to determine where the file resides. Because the disk is a mechanical device, this search is time consuming. By increasing the number of directory entries DOS keeps in memory, FASTOPEN increases the speed with which DOS locates files on disk, since it remembers the disk addresses of files you have already opened.

Format

```
FASTOPEN d:[=entries][...][/X]
```

d: Specifies the disk drive for which DOS is setting aside storage space for directory entries.

=entries Specifies the number of directory entries for which DOS is reserving space. This value must be in the range of 10 to 999. The default is 48.

/X A DOS 5 or later parameter that tells FASTOPEN to create the table of directory entries in expanded memory. Using /X limits the number of entries to 305.

Notes

Whenever DOS opens a file for reading or writing, it must locate that file on disk. DOS does this by first searching the directory specified (or the current directory) for the file's starting location.

Beginning with DOS 3.3, you can use FASTOPEN to set aside memory for a table of commonly used files and their starting locations. When you later open a file, DOS examines this table for the file's starting address. If DOS locates the file in the table, it doesn't have to read this information from disk. This saves considerable time.

The FASTOPEN command allows you to specify the size of the table of filenames DOS stores in memory. If several disks are specified, the sum of the number of directory entries cannot exceed 999 (305 if using the /X parameter).

If you are using a third-party disk defragmenter and FASTOPEN, always immediately reboot your system after the defragmenter completes.

NOTE

Example

In the following example, DOS remembers 50 directory entries for drive C:

```
C:\> FASTOPEN C:=50 <ENTER>
```

Messages

Cannot use FASTOPEN for drive *n*:

The disk drive specified is either a floppy disk drive or has been assigned or substituted. FASTOPEN supports fixed disks only.

FASTOPEN already installed

FASTOPEN has already been installed in memory. Because FASTOPEN installs memory-resident software, it can be invoked only once per user session. To change FASTOPEN parameters, you must restart DOS.

FASTOPEN EMS entry count exceeded

You specified more than 305 entries when using the /X parameter, or you do not have enough EMS available. Reduce the entry count and try again.

FASTOPEN installed - memory resident software

FASTOPEN has successfully loaded itself in memory and is ready to assist DOS in opening your files.

Invalid drive specified

One of the disk drives specified in the command line is invalid. Make sure that you are using existing disk drives followed by a colon.

FASTOPEN supports fixed disk drives only.

REMEMBER

Same drive specified more than once

You have placed the same disk drive letter twice in the FASTOPEN command line. Correct the error and reissue the command.

Too many drive entries

Invoke the FASTOPEN command with fewer disk drives. Specify only the drives you most commonly use.

Too many name entries

The total number of directory entries in the FASTOPEN command line exceeds 999. Reduce this number and reissue the command.

FC

Displays the actual differences between two files

FC stands for *file compare.* FC compares two files and displays those lines that differ.

Format

```
FC [/A][/B][/C][/L][/LB n][/N][/T][W][/nnnn]
  file1 file2
```

/A Tells FC to display only the first and last lines in a group of lines that differ, rather than all the lines that differ between two ASCII files.

/B Performs a binary comparison of the files.

/C Ignores the case of letters.

/L Performs an ASCII-file comparison.

/LB *n* Uses an internal buffer of *n* lines.

/N Displays the line numbers of lines that differ in two ASCII files.

/T Treats tabs as tabs, rather than expanding tabs into spaces.

/W Compresses white space (spaces or tabs).

/nnnn Specifies the number of lines that must again match following a difference before FC assumes the files are synchronized. The default is 2.

file1 The complete file specification of the first file to compare.

file2 The complete file specification for the second file to compare.

Notes

FC shows the differences between files by displaying the first filename, the line preceding the lines that don't match, then the lines that differ, and finally the line following the lines that don't match. Then the same information is provided for the second file.

FC compares and displays binary files in the form

```
aaaaaaaa bb cc
```

where aaaaaaaa is the offset address of the values that differ, and bb and cc are the values that differ.

If FC encounters too many errors, it terminates, displaying the following message:

```
Resynch failed. Files are too different.
```

Using FC to compare two files is often more helpful than using COMP because FC provides more information.

Examples

The following command compares two binary files, using the /B parameter:

```
C:\> FC /B TEST.EXE OLDTEST.EXE <ENTER>
```

In this case, the output might be

```
00000002: 9A 66
00000004: 06 20
00000006: 01 03
0000000A: 00 CE
0000000E: 20 14
0000000F: 00 04
00000011: 02 08
00000012: 32 97
```

Similarly, assuming that the files A.DAT and B.DAT contain the following,

A.DAT	B.DAT
4	4
5	5
A	B
A	B
A	B
6	6
7	7
8	8

the following command

```
C:\> FC A.DAT B.DAT <ENTER>
```

will display

```
***** A.DAT
5
A
A
A
6

***** B.DAT
5
```

```
B
B
B
6.
*****
```

If you include the /N parameter, FC will display

```
C:\> FC /N A.DAT B.DAT <ENTER>

***** A.DAT
    2:  5
    3:  A
    4:  A
    5:  A
    6:  6

***** B.DAT
    2:  5
    3:  B
    4:  B
    5:  B
    6:  6
*****
```

If you want FC to ignore differences between upper- and lowercase letters, add the /C parameter, as follows:

```
C:\> FC /C FILE.1 FILE.2 <ENTER>
```

Messages

Cannot open *FILENAME.EXT* - no such file or directory

FC could not locate the file you specified in the command. Check your spelling of the filename and current directory.

FILENAME.EXT larger than *FILENAME.EXT*

The contents of each file compared identically to a point, but the first file is longer than the second.

Incompatible switches

You have placed the /B (binary file switch) in the same command line as an ASCII-file parameter.

Insufficient number of filespecs

You used FC without specifying *file1* and *file2*.

No differences encountered

The files are identical.

Out of memory

Your system has insufficient RAM for FC to create the tables it needs to store information from each file during the comparison. If you have memory-resident software or a RAM drive installed, you might consider removing them.

Resynch failed
Files are too different

The files are so different that FC could not locate a position in both files similar enough to resume the file comparison.

FDISK

Defines disk partitions on a DOS fixed disk

DOS allows you to divide the fixed disk into logical collections of cylinders, called *partitions*. By so doing, you can place several operating systems into different partitions on one fixed disk. The FDISK command allows you to add, change, display, and delete disk partitions.

Format

```
FDISK
```

Notes

If your computer is new and DOS cannot locate the fixed disk, you must issue the FDISK command to define the disk to DOS. FDISK is a menu-driven program that allows you to divide your disks into multiple sections called *partitions*. In most cases, however, your computer dealer will have done this for you.

The first sector on any fixed disk contains a master boot record that contains information that defines the partition the computer uses for booting. FDISK is your means of interfacing with the master boot record.

Example

To use FDISK, place your DOS disk in the drive and type

```
C:\> A: <ENTER>
A:\> FDISK <ENTER>
```

FDISK, a menu-driven program, responds with its first menu:

```
       MS-DOS Version 6.00
     Fixed Disk Setup Program
  (C)Copyright Microsoft Corp. 1983-1993
          FDISK Options


Current fixed disk drive: 1
Choose one of the following:
1: Create DOS Partition or Logical DOS Drive
2: Set active partition
3: Delete DOS Partition or Logical DOS Drive
4: Display partition information

Enter choice:[1]

Press Esc to exit FDISK
```

Messages

DOS partition created

FDISK has successfully created your DOS partition as specified. Once you exit FDISK, you must format the disk before it can be used by DOS.

Do you wish to use the entire fixed disk for DOS (Y/N).....? [Y]

FDISK is prompting you as to whether or not you want to use the entire disk as a DOS partition. Most users will want to use their entire disk as a DOS partition.

Do you wish to use the maximum size *nnn* for a DOS partition and make the DOS partition active (Y/N)...?

FDISK is prompting you as to whether or not you want to use all of the remaining disk cylinders as a DOS partition. Most users will want to do so.

Enter the number of the partition you want to make active.....[*n*]

FDISK is prompting you for the number associated with the disk partition that you want to make active. Although FDISK lets you logically divide your disk into several partitions, only one partition can be active (or the partition your computer uses to start DOS). Mostly, you will keep your DOS partition as the active partition.

Fixed disk already has a DOS partition

You have selected the Create DOS Partition when a DOS partition already exists. You can only place one active DOS partition on a hard disk.

Maximum space available for partition *n* is *nnn* cylinders

You have instructed FDISK not to use the entire disk for the partition you are creating. FDISK is telling you the maximum size of the partition that you can create.

No DOS partition to delete

You have selected the FDISK menu option Delete DOS partition and the disk does not contain a DOS partition that DOS can delete.

No fixed disk present

You must inform your computer's CMOS about your hard disk type. If you are using an IBM PC AT or AT-compatible, use the SETUP program that accompanied your computer on floppy disk. If you are using a 386 or 486 system, you typically invoke SETUP by pressing a key such as DEL when you first start your computer. Refer to the documentation that accompanied your computer for more information on SETUP.

No partitions defined

You have selected the Display Partition Information option and you have not yet created partitions on the fixed disk.

No partition to make active

You have selected the Change Active Partition option and you have not yet created partitions on the fixed disk that FDISK can make active.

No space to create a DOS partition

You have selected the Create DOS Partition option and your disk does not have sufficient available free cylinders to create a DOS partition. If this error occurs, you may have to modify the size of the non-DOS partitions you have created on the disk.

Partition *n* is already active

You have selected the Change Active Partition option and the partition that you specified was already active.

Partition *n* made active

You have selected the Change Active Partition option and specified a new boot partition. This message is purely informational.

System will now restart
Insert DOS diskette in drive A
Press any key when ready...

You have successfully created a DOS partition and FDISK is ready to restart your system so that you can format your newly created fixed disk. Place a bootable disk in drive A and press any key to continue.

Total disk space is *nnn* cylinders
Maximum available space is *nnn* cylinders at *nn*

FDISK is telling you the current disk utilization and maximum space that you can use to create an additional disk partition.

Warning! Data in the DOS partition will be lost. Do you wish to continue.....? [N]

FDISK is prompting you to specify whether or not you really want to delete the current DOS partition, warning you that if you do so, all of the data that the partition contains will be lost. If you simply press ENTER, FDISK will select the default option of No.

Warning! Data in the Extended DOS partition will be lost. Do you wish to continue.....? [N]

FDISK is prompting you to specify whether or not you really want to delete an extended DOS partition, warning you that if you do so, all of the data that the partition contains will be lost. To delete the extended DOS partition, type Y and press ENTER. If you don't want to delete the extended DOS partition, simply press ENTER and FDISK will select the default of No.

Cannot FDISK with network loaded

FDISK is telling you that you cannot execute the FDISK command on disk that is part of a local area network.

Requested partition size exceeds the maximum available space

The number of cylinders that you have directed FDISK to use for your disk partition exceeds the number of available cylinders. Repeat the option, specifying a smaller number of cylinders.

Partition selected is not bootable active partition not changed

You have selected the Change Active Partition option and specified a partition that cannot be booted. FDISK has left the active partition unchanged.

Cannot create a zero size partition

The partition size you specified was 0Mb or 0%. The minimum partition size FDISK supports is 1Mb or 1%.

Cannot create an extended DOS partition without a primary DOS partition on disk 1

FDISK is informing you that you cannot create an extended DOS partition on a disk drive until you create a primary (bootable) DOS partition on the same disk. The primary partition is the disk partition from which your computer will boot DOS.

Cannot create Logical DOS drive without an Extended DOS Partition on the current drive

You tried to define a logical disk drive before creating an extended DOS partition within which the logical drive will reside. After you use FDISK to create an extended DOS partition, FDISK lets you divide the partition in up to 23 logical drives.

Cannot delete Extended DOS Partition while logical drives exist

You have attempted to remove the extended DOS partition without first individually removing the logical drives it contains. To remove the extended partition, you must first use FDISK to remove the logical drives.

All available space in the Extended DOS Partition is assigned to logical drives

There is no more disk space available to create additional logical disk drives.

All logical drives deleted in the Extended DOS Partition

FDISK has removed all the extended DOS partition's logical drives. You can now delete or change the size of the extended partition.

Extended DOS Partition already exists

You attempted to create an extended DOS partition and one is already defined. You can only create one primary and one extended DOS partition on your disk.

Extended DOS Partition created

FDISK successfully created the extended DOS partition using the size specified. You must now divide the extended partition into logical disk drives.

Extended DOS Partition deleted

FDISK has removed the extended partition. If you are using DOS 4 or DOS 5, change your primary DOS partition to use the space previously used by the extended partition.

No Logical DOS Drive(s) to delete

You have directed FDISK to delete a logical drive from the extended DOS partition, and no logical drives are defined.

Partition selected is not bootable, active partition not changed

You tried to change the active partition (the partition from which your computer boots DOS) to a partition that is not bootable. FDISK did not change the partition.

FIND

Searches file(s) or piped input for a character string

The FIND command lets you locate a sequence of characters or words within a file or within redirected output. FIND can also be used as a filter with piped input.

Format

```
FIND[/C][/I][/N][/V] "string"[file_spec][...]
```

/C Tells FIND to display a count of occurrences of the string. If /C is used with /N, the /N parameter is ignored.

/I A DOS 5 or later switch that directs FIND to consider upper- and lowercase letters as the same.

/N Tells FIND to precede each line containing the string with that line's number.

/V Tells FIND to display each line that does not contain the string.

string Specifies the string FIND is to search for. This string must be enclosed in quotation marks.

file_spec The name of the file in which to search for the string. This can be a series of filenames, separated by spaces. FIND does not support DOS wildcard characters.

Notes

The string FIND is searching for must be enclosed in quotation marks. If the string includes quotation marks as characters, you must use two sets of quotation marks—one set for the string and one for the command, as follows:

```
C:\> FIND """Look"" he said" FILENAME.EXT <ENTER>
```

Prior to DOS 5, command-line parameters had to come between FIND and the string to be found. In DOS 5, they can occur in any order.

Examples

In the following example, FIND is used as a filter to list each subdirectory in the current directory. If the current directory contains the following information:

```
C:\> DIR <ENTER>

 Volume in drive C is MSDOS
 Volume Serial Number is 1931-9E01
 Directory of C:\SUBDIR

.            <DIR>     01-20-93   6:44p
..           <DIR>     01-20-93   6:44p
ONE          <DIR>     01-20-93   6:44p
FILENAME EXT         6 01-20-93   6:45p
TWO          <DIR>     01-20-93   6:45p
READ     ME          4 01-20-93   6:45p
NOTES    DOC         3 01-20-93   6:45p
THREE        <DIR>     01-20-93   6:45p
        8 file(s)          13 bytes
                     85256192 bytes free
```

you can use the FIND command as follows to display directory names only:

```
C:\> DIR | FIND "<DIR>" <ENTER>
.            <DIR>     01-20-93   6:44p
..           <DIR>     01-20-93   6:44p
ONE          <DIR>     01-20-93   6:44p
TWO          <DIR>     01-20-93   6:45p
THREE        <DIR>     01-20-93   6:45p
```

To list all files that are not directories, use the FIND /V parameter, as follows:

```
C:\> DIR A: | FIND /V "<DIR>" <ENTER>
```

The following command displays each occurrence of the string "begin" in the TEST.PAS file:

```
C:\> FIND "begin" TEST.PAS <ENTER>
```

The following command also displays each occurrence of "begin". In this example, however, each line is preceded by its line number:

```
C:\> FIND /N "begin" TEST.PAS <ENTER>
```

The following command displays the number of occurrences of "begin" that are in the file:

```
C:\> FIND /C "begin" TEST.PAS <ENTER>
```

Messages

FIND: Access denied *FILENAME.EXT*

FIND could not access the file specified because it is locked by another file in the network.

FIND: Read error in *FILENAME.EXT*

The file specified probably is corrupted. Repeat the command to see whether the error goes away. If you don't have a backup copy of the file, you may have to use the RECOVER command to save a portion of the file.

FIND: Syntax error

The FIND command format is invalid. Make sure that you specify the search string and parameters in the right order.

FOR

Provides repetitive execution of DOS commands

The FOR command is used most commonly within DOS batch files to repeat a specific command for a given set of files.

Format

```
FOR %%variable IN (set) DO DOS_command
```

%%variable The FOR-loop control variable that DOS manipulates with each iteration. The variable name can be any single printable character except %, =, |, /, <, >, or a comma.

set A list of valid DOS filenames. This list can contain DOS filenames separated by commas, wildcard characters, or both. The list must be enclosed in parenthesis.

DOS_command The command to be executed with each iteration.

Notes

The FOR command can be used from the DOS prompt. Use the %% symbols before the variable name when working within batch files. Use the single % from the DOS prompt.

Examples

The following shows how FOR works in a batch file:

```
FOR %%V IN (ONE.BAT, TWO.DAT, THREE.INI) DO TYPE %%V
```

In this case, FOR assigns the variable filename ONE.BAT during the first iteration of the command and displays the contents of the file:

```
TYPE ONE.BAT
```

On the second iteration, FOR assigns the variable filename TWO.DAT and displays the contents of the file:

```
TYPE TWO.DAT
```

On the third iteration, FOR assigns the variable filename THREE.INI, and again displays the contents of the file:

```
TYPE THREE.INI
```

When FOR prepares for the fourth iteration, it fails to find any additional filenames. Processing is complete.

Messages

FOR cannot be nested

You have placed a second FOR command as the command FOR will execute with each iteration. This is called *nesting*, and is not allowed.

Syntax error

The command format is incorrect. Make sure that the command line contains the word "DO."

FORMAT

Formats a disk for use by DOS

The manufacturers of floppy disks have no way of knowing on what computer—or operating system—the disks will be used. Therefore, before you can use a new disk, you must format it for DOS. The FORMAT command lets you format a disk that is to be used with DOS.

Format

```
FORMAT d:[/V:label][/S][/B][/Q]
FORMAT d:[/F:size][/U][/S][/B][/Q]
FORMAT d:[/1][/4][/U][/V:label][/S][/B][/Q]
FORMAT d:[/8][/U][/S][/B][/Q]
FORMAT d:[/T:tracks][/N:sectors][/S][/B][/Q]
```

/1 Tells FORMAT to format the disk as a single-sided disk.

/4 Tells FORMAT to format the disk as double-sided in a quadruple-density disk drive.

/8 Tells FORMAT to format the disk with 8 sectors per track; most disks use 9 or 15 sectors.

/N:sectors Defines the number of sectors to a track. This parameter requires DOS 3.3 or later.

/T:tracks Defines the number of tracks to a side. This parameter requires DOS 3.3 or later.

/F:size Beginning with DOS 4, specifies the size of the floppy disk to format, where size is one of the following:

160	160K	160Kb			
180	180K	180Kb			
320	320K	320Kb			
360	360K	360Kb			
720	720K	720Kb			
1200	1200K	1200Kb	1.2	1.2M	1.2Mb
1440	1440K	1440Kb	1.44	1.44M	1.44Mb
2880	2880K	2880Kb	2.88	2.88M	2.88Mb

/S Tells FORMAT to place the system files on the disk, thereby making the disk bootable.

/B Tells FORMAT to reserve space for the system files on the target disk. Unlike the /S parameter, /B does not actually place the files on disk. The /S and /V parameters cannot be used in conjunction with the /B parameter.

/V:label Tells FORMAT to include the volume label specified.

/U Unconditional format (DOS 5); destroys all information previously recorded on a disk. You cannot unformat an unconditionally formatted disk.

/Q Performs a *quick* format of a previously formatted disk (DOS 5 or later), overwriting the root directory and file allocation table. A quick format does not search the disk for bad sectors.

Notes

Inadvertently formatting a fixed disk can be disastrous. FORMAT first prompts you as follows:

```
WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE N: WILL BE LOST!
Proceed with Format (Y/N)?
```

To proceed with the formatting process, type **Y**; otherwise, type **N**. Depending on which version of DOS you are using, FORMAT may also prompt you to enter the volume label of the hard disk before formatting. If the label you enter does not match the hard disk label, FORMAT will not continue. When the formatting process is complete, FORMAT displays the following:

- Total disk space
- Corrupted disk space marked as defective
- Total disk space consumed by the operating system
- Total disk space available for file utilization

FORMAT reports on defective space it finds during formatting. FORMAT also places entries for each defective sector into the file allocation table (an area that DOS uses to keep track of the locations of files). This prevents DOS from using the corrupted sectors for data storage.

The FORMAT /S command copies the hidden files that are needed in order to boot DOS to the target disk.

Do not use FORMAT in conjunction with ASSIGN, JOIN, or SUBST.

FORMAT will not work with drives that are being used in a network configuration.

If you are using DOS 5 or later, you may be able to recover an inadvertently formatted hard disk by using the UNFORMAT command.

FORMAT supports the following exit status codes:

Exit Status	Meaning
0	Successful format
3	Format terminated by user with CTRL-C
4	Fatal error
5	User typed N at the prompt to continue

Examples

This command formats the disk in drive A as bootable:

```
C:\> FORMAT A:/S <ENTER>
```

When invoked, the command will display the following:

```
Insert a new diskette for drive A:
and press ENTER when ready.
```

Many users frequently have to format double-density disks in their 1.2Mb drives. Use the /4 parameter to tell FORMAT to create a 360Kb disk:

```
C:\> FORMAT A: /4 <ENTER>
```

The /B and /S parameters are similar. The following command tells FORMAT to reserve space for the operating system's boot files (files that are necessary for the operating system to boot), instead of placing those files on disk:

```
C:\> FORMAT A: /B <ENTER>
```

If you include the /B parameter, the SYS command can later update the disk as required.

If you are using DOS 5 or later, the following command tells FORMAT to perform a quick format of a previously formatted floppy disk that currently resides in drive A:

```
C:\> FORMAT A: /Q <ENTER>
```

Messages

Attempted write-protect violation

The disk in the target drive is write-protected.

Bad partition table

The partition table DOS uses to keep track of the fixed disk's layout is bad. Invoke the FDISK command to correct the partition table layout.

Cannot find System files

The FORMAT command line contains the /S parameter, which tells FORMAT to create a bootable target disk, but FORMAT could not locate the hidden system files it must copy to the target disk in order to do so. Find the disk you used to boot DOS, and use it in the FORMAT operation.

Disk unsuitable for system disk

The first portion of the disk specified, which DOS uses to store its system files, contains damaged sectors. Since DOS cannot place these files on the disk at this location, FORMAT cannot create a bootable disk.

Drive letter must be specified

You did not specify a target disk drive. Reissue the FORMAT command, specifying the drive that contains the disk to be formatted.

Enter current Volume Label for drive *n*:

FORMAT is prompting you to enter a disk volume name; it will assign this name to the disk on completion of the format operation. Type an 11-character disk volume name, or press ENTER for no name.

Error reading partition table

FORMAT encountered an error while attempting to read the partition table. Restart DOS and repeat the command. If this error persists, invoke the FDISK command to correct the partition-table error.

Error writing directory

FORMAT could not create a root directory on the disk specified. This error is normally associated with an unusable disk.

Format another (Y/N)?

FORMAT has completed processing one disk and is asking whether you want to format additional disks. If you do, type **Y** and press ENTER; FORMAT will prompt you to insert another disk. Otherwise, type **N** and press ENTER; FORMAT will return control to DOS.

Format complete

The format operation was successful.

Insert DOS diskette for drive *n* and strike ENTER when ready

The FORMAT command line contains the /S parameter, which directs FORMAT to create a bootable DOS disk. In this case, FORMAT is prompting you to insert a bootable DOS disk in the drive specified so that it can copy the hidden files to the target disk.

Insert new diskette for drive *n*: and strike ENTER when ready

FORMAT is prompting you to place the disk you want to format in the drive specified. After you do so, press the ENTER key; FORMAT will continue its processing.

Invalid characters in volume label

The FORMAT command line contains the /V parameter that tells FORMAT to assign a disk volume name to the disk. One or more of the letters you entered were invalid for a disk volume label.

Invalid media or Track 0 bad - disk unusable

The first track on the disk specified is damaged. DOS must be able to format the first track of a disk, either to make the disk bootable or to place in that track information that tells DOS that the disk is not a bootable system disk. If DOS cannot use this track, DOS cannot use the disk.

***n* bytes total disk space**
***n* bytes used by system**
***n* bytes in each allocation unit**
***n* allocation units on disk**

FORMAT is simply giving you information about your newly formatted disk.

***n* bytes in bad sectors**
***n* bytes available on disk**

FORMAT is providing a summary of disk usage for the newly formatted disk.

Non-System disk or disk error
Replace and strike any key when ready

The FORMAT command line contains the /S parameter, which tells FORMAT to create a bootable disk. In this case, either the disk in the drive specified did not contain the hidden system files, or FORMAT could not read the disk. Place the disk you used to boot DOS in the drive and press any key to continue.

Parameters not compatible

Two or more parameters in the FORMAT command line cannot be issued in conjunction with one another. Review the format of the FORMAT command and reissue the command.

Parameters not supported by drive *n*:

One or more parameters in the FORMAT command line is invalid for the disk drive specified. Review the parameters and reissue the command.

System transferred

The FORMAT command line contains the /S parameter, which tells FORMAT to build a bootable disk. FORMAT has successfully copied the hidden system files to the target disk.

Unable to write BOOT

FORMAT could not write to the first track of the disk the information FORMAT uses to create a bootable disk. Repeat the FORMAT command to see whether the error corrects itself. If not, the disk is unusable.

Volume label (11 characters, ENTER for none)?

The FORMAT command line contains the /V parameter, which tells FORMAT to prompt you for a disk volume name. Type an 11-character disk volume name, or press ENTER for no name.

WARNING, ALL DATA ON NON-REMOVABLE
DRIVE *n*: WILL BE LOST
Proceed with Format (Y/N)?

FORMAT is warning you that the disk specified in the FORMAT command line is a fixed disk, and that continuing with the FORMAT operation will destroy all information on that disk. If you want to proceed with the format operation (overwriting the information on the fixed disk), type Y and press ENTER. Otherwise, type N and press ENTER.

GOTO

Branches to the label specified in a batch file

The GOTO command tells DOS to branch to a label specified in a batch file. Label names contain any of the characters valid for DOS filenames. If the label does not exist, DOS terminates execution of the batch file.

Format

```
GOTO label_name
```

label_name Specifies the name of a label within a DOS batch procedure.

Notes

DOS label names can be virtually any length. However, DOS distinguishes only the first eight characters of a label name. It will consider the label names DOS_LABEL1 and DOS_LABEL2 to be equivalent, because the first eight characters of each name are equivalent.

In the batch file, a colon must precede the label.

If the label specified in a GOTO command does not exist or is not preceded by a colon, DOS will terminate the batch file and display an error message.

Example

The following batch procedure repeatedly displays a directory listing of the current drive. It runs until the user presses CTRL-C or CTRL-BREAK:

```
:LOOP
DIR
GOTO LOOP
```

Message

Label not found

The label following the GOTO command was not found in the batch file.

GRAFTABL

Improves extended character display in graphics mode

The GRAFTABL command allows you to display extended ASCII characters when the display is in medium-resolution graphics mode. DOS 6 does not support GRAFTABL.

Format

```
GRAFTABL[code_page | /STATUS | /?]
```

code_page Specifies the code page to be used for the display. Possible values are as follows:

Code Page	Country
437	United States
850	Multilingual
852	Slavic (DOS 5 or later)
860	Portuguese
863	French Canadian
865	Nordic

/STATUS Tells GRAFTABL to display the code page currently in use.

/? Tells GRAFTABL to display the code page plus a list of available options.

Notes

Most users will not have to use the GRAFTABL command. If you do, you probably should place the GRAFTABL command in your AUTOEXEC.BAT file.

GRAFTABL is a memory-resident program that supports the following exit status values:

Exit Status	Meaning
0	Table successfully loaded
1	Table replaced an existing table
2	File error preventing loading of a new table
3	Invalid command-line parameter
4	Incorrect DOS version

Examples

If you specify the /STA parameter in the GRAFTABL command line, as follows:

```
C:\> GRAFTABL /STATUS <ENTER>
```

GRAFTABL displays the current code page, as shown here:

```
USA version of Graphic Character Set Table is already loaded.
```

The following command:

```
C:\> GRAFTABL 437 <ENTER>
```

tells GRAFTABL to use the code page for the United States when displaying extended characters.

Message

***nnnn* version of Graphic Character Set Table is already loaded**

You have invoked GRAFTABL a second time. It needs to be invoked only once per user session.

GRAPHICS

Provides print-screen support for graphics mode

The GRAPHICS command lets you use print-screen operations to print contents of screens that contain graphics.

Format

```
GRAPHICS [pro_file][printer_type][/B][/LCD][/R][/PRINTBOX:id]
```

pro_file The path name of a file that contains the profiles of supported printers. By default, DOS looks for GRAPHICS.PRO. This parameter requires at least DOS 4.

printer_type Specifies the target printer type. Possible values are as follows:

Value	Meaning
COLOR1	Color printer with black ribbon
COLOR4	Color printer with RGB ribbon

Value	**Meaning**
COLOR8	Color printer with cyan, magenta, yellow, and black ribbon
GRAPHICS	Graphics printer
GRAPHICSWIDE	Graphics printer with 11-inch carriage (DOS 4 or later)
LASERJETII	Hewlett-Packard Laserjet II (DOS 5 or later)
PAINTJET	Hewlett-Packard Paintjet (DOS 5 or later)
QUIETJET	Hewlett-Packard Quietjet (DOS 5 or later)
QUIETJETPLUS	Hewlett-Packard Quietjet Plus (DOS 5 or later)
RUGGEDWRITER	Hewlett-Packard Rugged Writer (DOS 5 or later)
RUGGEDWRITERWIDE	Hewlett-Packard Rugged Writer Wide (DOS 5 or later)
THERMAL	Thermal printer
THINKJET	Hewlett-Packard Thinkjet (DOS 5 or later)

/B Tells GRAPHICS to print the background color. The default is not to print the background color.

/LCD Directs GRAPHICS to use the LCD aspect ratio. The /LCD switch is identical to /PRINTBOX:LCD.

/R Tells GRAPHICS to reverse the color of the screen image—black images on the screen are printed as black, white images as white.

/PRINTBOX:id Selects the print box size. The following values are valid: STD for standard and LCD for liquid crystal display.

Notes

The GRAPHICS command invokes a program that remains in memory until the computer is rebooted. Therefore, this command needs to be invoked only once during a user session.

If you use the GRAPHICS command to print graphics images to the printer, and the printouts are too dark, use the /R parameter.

Examples

The following command loads the memory-resident software required to support print-screen operations that contain graphics images:

```
C:\> GRAPHICS <ENTER>
```

The following command tells DOS to print white screen images as black, and black screen images as white:

```
C:\> GRAPHICS /R <ENTER>
```

Message

Unrecognized printer

The printer type you have specified in the command is invalid. Make sure that your printer type is on the list of valid printers.

HELP

Starts the DOS 5 online help facility

Starting with version 5, DOS provides onscreen help text that describes the DOS commands and their formats. The command to get help in DOS 5 is HELP, optionally followed by the command for which you want information; in DOS 6 it is FASTHELP. You can also get help about a command by typing the command name, followed by the /? switch. See FASTHELP for more details.

Format

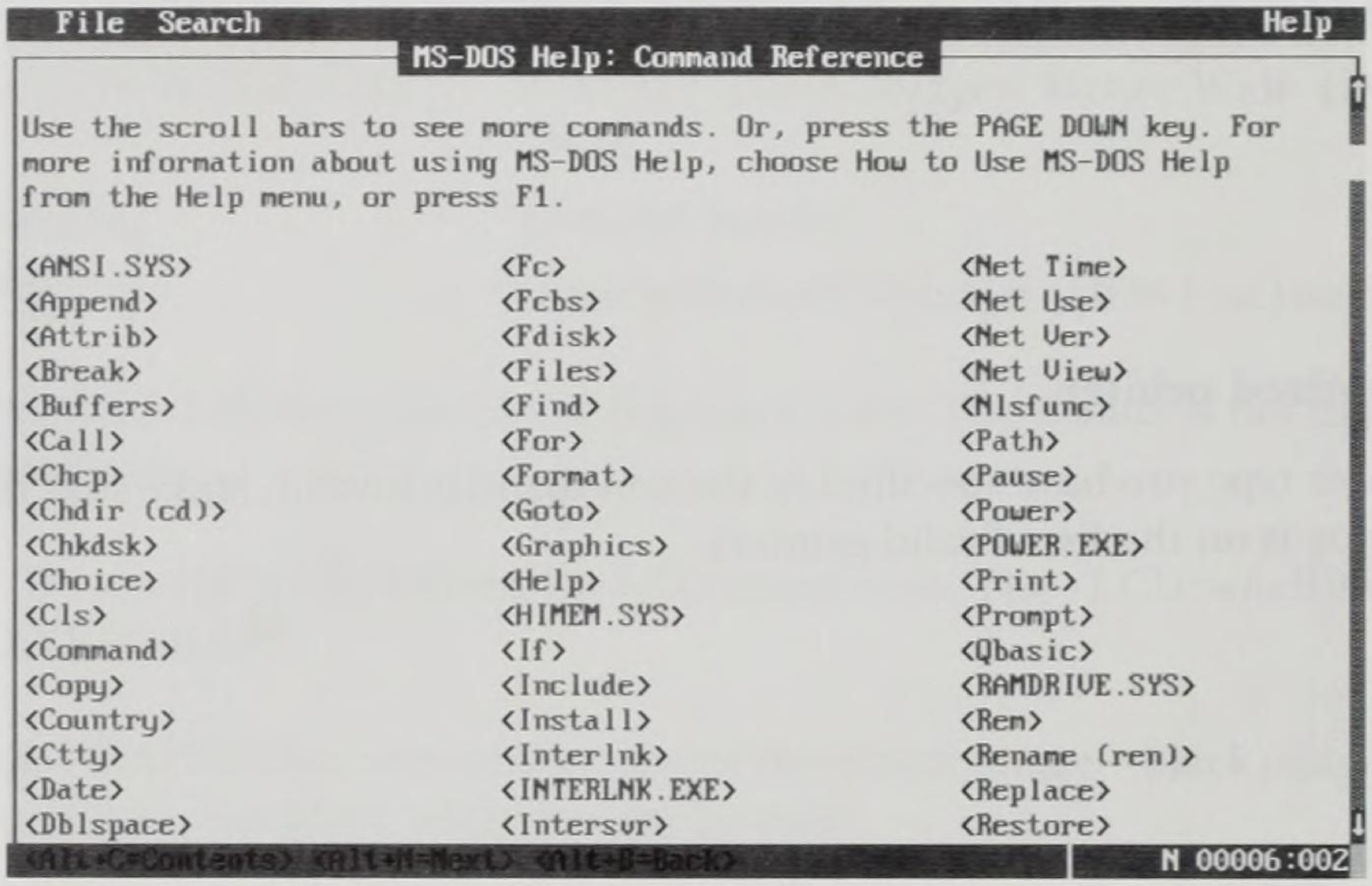
```
HELP [CommandName]
```

Notes

DOS 6 provides an interactive online help facility that provides descriptions, examples, and explanations of different DOS commands. After you invoke HELP, you can use the arrow keys or the mouse to select and display information on different commands.

Example

When you invoke HELP without specifying a command, HELP displays a list of the DOS commands as shown here:


File
Search
Help
MS-DOS Help: Command Reference
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
N 00006:002

```
Use the scroll bars to see more commands. Or, press the PAGE DOWN key. For
more information about using MS-DOS Help, choose How to Use MS-DOS Help
from the Help menu, or press F1.

<ANSI.SYS>                 <Fc>                      <Net Time>
<Append>                   <Fcbs>                    <Net Use>
<Attrib>                   <Fdisk>                   <Net Ver>
<Break>                    <Files>                   <Net View>
<Buffers>                  <Find>                    <Nlsfunc>
<Call>                     <For>                     <Path>
<Chcp>                     <Format>                  <Pause>
<Chdir (cd)>               <Goto>                    <Power>
<Chkdsk>                   <Graphics>                <POWER.EXE>
<Choice>                   <Help>                    <Print>
<Cls>                      <HIMEM.SYS>               <Prompt>
<Command>                  <If>                      <Qbasic>
<Copy>                     <Include>                 <RAMDRIVE.SYS>
<Country>                  <Install>                 <Rem>
<Ctty>                     <Interlnk>                <Rename (ren)>
<Date>                     <INTERLNK.EXE>            <Replace>
<Dblspace>                 <Intersvr>                <Restore>
```

The following HELP command displays a screenful of information on the XCOPY command:

```
C:\> HELP XCOPY <ENTER>
```

Message

Cannot find the file QBASIC.EXE

To use the HELP command, the DOS file QBASIC.EXE must reside in the current directory or a directory specified in the command path.

IF

Provides conditional processing within DOS batch files

The IF command lets your batch files test whether a specific file exists, whether two character strings are equal, or whether the preceding command was successful.

Format

```
IF [NOT] condition DOS_command
```

NOT Performs a Boolean NOT on the result of condition.

condition Must be one of the following:

Condition	**Meaning**
ERRORLEVEL value	True if the preceding program's exit status is greater than or equal to the value
EXIST file_spec	True if the file specified exists
string1==string2	True if the two strings are identical

DOS_command The name of the DOS command it is to perform if the condition is true.

Notes

Be careful when you use IF to test a program's exit status value. IF evaluates to true if the exit status is greater than or equal to the value specified. If you combine two IF commands, you can determine the exact exit status value:

```
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO One
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO Two
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO Three
```

If you use IF to compare two strings, you should group the strings within single quotation marks. The IF command requires two strings to compare. If only one string is present, IF displays a Syntax error message.

If the user does not specify a value for %1 in the following example, IF will compare the null string '' to the string 'CONFIG.SYS':

```
IF '%1'=='CONFIG.SYS' TYPE %1
```

If the single quotation marks are not present, IF has only one string to compare and generates an error.

Although most users use the IF statement only from within batch files, IF is fully supported from the command line.

If you are using DOS 6, see the CHOICE command.

Examples

The following batch file uses IF to determine whether CONFIG.SYS exists in the current directory and, if so, copies it to drive A. Note the use of NOT to determine whether the file does not exist:

```
@ECHO OFF
IF EXIST CONFIG.SYS COPY CONFIG.SYS A:
IF NOT EXIST CONFIG.SYS ECHO File not found
```

Here, if the DOSPGM program exits with a status greater than or equal to 3, a message to that effect is displayed:

```
@ECHO OFF
DOSPGM
IF ERRORLEVEL 3 ECHO T H R E E
```

The following example uses the IF command to determine whether the value of a batch parameter is defined:

```
@ECHO OFF
IF '%1'=='' ECHO Not defined
IF NOT '%1'=='' ECHO The value is %1
```

Messages

Bad command or filename

The command specified after the condition in the command line does not exist as specified.

Syntax error

The format of the command is invalid.

INTERLNK

Redirects requests for a client drive or printer to a server

The DOS 6 INTERLNK and INTERSVR commands allow two computers to easily exchange files or share disks or printers over a serial or parallel cable. These two commands provide an ideal way for a user with a laptop computer to upload files to or retrieve files from a desktop PC.

Format

```
INTERLNK [Client[:]=[Server[:]]
```

Client Specifies the local drive letter with which you want to refer to a specific disk drive on the server.

Server Specifies the server disk you want to access.

Notes

Before you can use the INTERLNK command, you must first install the INTERLNK.EXE device driver in your CONFIG.SYS file. The format of the DEVICE entry is as follows:

```
DEVICE=C:\DOS\INTERLNK.EXE [/DRIVES:NumberOfDrives]
        [/NOPRINTER][/AUTO][/BAUD:Rate][/LOW]
        [/NOSCAN][/V][/COM[:][Number|Address]]
        [/LPT[:][Number|Address]]
```

/DRIVES:NumberOfDrives Specifies the number of remote disk drives INTERLNK will redirect. The default is 3. If you specify the value 0, INTERLNK will only redirect printers.

/NOPRINTER Directs INTERLNK not to redirect remote printers.

/AUTO Directs INTERLNK only to install its device driver if a connection can immediately be established with a remote computer.

/BAUD:Rate Specifies the data communication rate for a serial connection. Valid rates include 9600, 19200, 38400, 57600, and 115200. The default is 115200.

/LOW Directs INTERLNK to load its driver into conventional memory as opposed to upper memory.

/NOSCAN Directs INTERLNK to install its device driver but not to scan for a connected remote computer.

/V Prevents INTERLNK conflicts with the computer's built-in timer. Use this switch if a computer quits running during a serial connection.

/COM[:][Number | Address] Specifies the COM port or address through which a serial connection will be made.

/LPT[:][Number | Address] Specifies the LPT port or address through which a parallel connection will be made.

To connect the computer, you must use a NULL-modem cable if you are using serial ports, or a bidirectional parallel cable if you are using parallel ports.

INTERLNK requires DOS 6.

Examples

The following CONFIG.SYS entry loads the INTERLNK device driver to support data exchange over a parallel cable connected to LPT1:

```
DEVICE=C:\DOS\INTERLNK.EXE /LPT1 /DRIVES:5
```

The following command directs INTERLNK to route references to the local (client) disk drive F to the server drive C:

```
C:\> INTERLNK F:=C: <ENTER>
```

When you later refer to drive F, INTERLNK will route the request to drive C. To view the current mappings, invoke INTERLNK as follows:

```
C:\> INTERLNK <ENTER>

       Port=COM1

       This Computer        Other Computer
         (Client)              (Server)
       -------------   -------------------------
        D:    equals    A:
        E:    equals    B:
        F:    equals    C: (499Kb)
        LPT2: equals    LPT1:
        LPT3: equals    LPT2:
```

As you can see, INTERLNK also displays available server printers. In this case, you can access the server's LPT1 printer using the client port name LPT2.

If you want to disable disk reference, invoke INTERLNK with the drive letter followed by an equal sign as shown here:

```
C:\> INTERLNK F= <ENTER>
```

INTERSVR

Loads server that allows remote disk and printer access

The DOS 6 INTERSVR command works with the INTERLNK command to allow two computers to quickly exchange files over a serial or parallel cable connection.

Format

```
INTERSVR [Drive:] [/X=Drive:] [/LPT:[Number | Address]]
         [/COM:[Number | Address][/BAUD:Rate] [/B] [/V] [/RCOPY]
```

Drive Specifies a specific server drive you want available for client users. If you don't specify a drive, INTERSVR makes all drives available.

/X=Drive Specifies a server drive you want to exclude client users from accessing.

/LPT:[Number | Address] Specifies the LPT port number or address through which the connection will be make.

/COM:[Number | Address] Specifies the COM port or address through which the connection will be made.

/BAUD:Rate Specifies the data communication rate for a serial connection. Valid rates include 9600, 19200, 38400, 57600, and 115200. The default is 115200.

/B Directs INTERSVR to display a black-and-white screen.

/V Prevents INTERSVR conflicts with the computer's built-in timer. Use this switch if a computer quits running during a serial connection.

/RCOPY Directs INTERSVR to copy the INTERLNK and INTERSVR files to a remote client over serial lines.

Notes

The INTERSVR command does not require a device driver; to create a server, you simply invoke INTERSVR.

See INTERLNK. INTERSVR requires DOS 6.

Examples

The following command makes all the server's drives available for redirection and specifies the connection between computers will occur using LPT1:

```
C:\> INTERSVR /LPT1 <ENTER>
```

In a similar way, the next command specifies the connection will occur over COM1, but it excludes the redirection of drive D:

```
C:\> INTERSVR /COM1 /X=D <ENTER>
```

If the client computer to which you want to connect does not contain the INTERLNK and INTERSVR commands, you can copy the files to the client using a NULL-modem cable and serial ports. The following command directs INTERSVR to copy the files to the client:

```
C:\> INTERSVR /RCOPY <ENTER>
```

After you issue this command, follow the prompts INTERSVR displays.

Message

The are no *xxxx* ports available for remote installation

INTERSVR could not find a serial or parallel port for use in a remote computer connection.

JOIN

Joins a disk drive to an empty subdirectory

JOIN makes two disks seem like one by joining a disk to a DOS path that represents an empty subdirectory. DOS 6 does not support the JOIN command.

Format

```
JOIN[d1:[d2:path]][/D]
```

d1: Specifies the disk drive to be joined to the path provided.

d2:path Specifies the directory to be joined.

/D Disconnects a previously joined disk.

Notes

Users with two floppy-drive systems often have applications that must open more files than can fit on a single disk. By using the JOIN command, they can "join" the disk in drive B to a subdirectory in drive A, making the two disk drives seem to be one large single disk.

If you issue a JOIN command without any parameters, JOIN displays the current joins. DOS will join a disk only to an empty DOS directory.

Do not use JOIN in conjunction with the BACKUP, CHKDSK, DISKCOMP, DISKCOPY, FDISK, FORMAT, LABEL, RECOVER, RESTORE, or SYS commands.

Examples

The following command joins drive A to the empty subdirectory JOINDIR, which resides on drive C:

```
C:\> JOIN A: \JOINDIR <ENTER>
```

In this case, references to C:JOINDIR are identical to references to drive A. If drive A contains subdirectories, simply refer to them as follows:

```
C:\> DIR \JOINDIR\SUBDIR <ENTER>
```

To remove a join, use the /D parameter:

```
C:\> JOIN A: /D <ENTER>
```

Message

Directory not empty

You are attempting to join a disk to a DOS subdirectory that contains other subdirectories or files. DOS requires that the join directory be empty prior to the JOIN operation.

KEYB

Loads a foreign keyboard set

To fully support international configurations, DOS provides support for various keyboard templates. The KEYB command loads memory-resident software to replace the standard keyboard layout provided by DOS.

Format

```
KEYB[keyboard_code[,code_page],[filespec]]]
  [/E][/ID:keyboard]
```

keyboard_code Specifies the two-letter code associated with the desired keyboard.

code_page The desired code page (see CHCP).

filespec The name of the file containing the keyboard definitions (normally, KEYBOARD.SYS).

/E Setup for an enhanced keyboard (DOS 5 or later).

/ID:keyboard Specifies the keyboard type (DOS 5 or later); only needed for countries with two or more layouts.

Notes

Once a new keyboard is installed, you can toggle between it and the default keyboard by pressing CTRL-ALT-F1 for the default keyboard and CTRL-ALT-F2 for the foreign keyboard. Common keyboard layouts (and the matching keyboard codes) include

Keyboard Code	Country
BE	Belgium
BR	Brazil
CF	Canadian French
CZ	Czech
DK	Denmark
FR	France
GR	Germany
HU	Hungary
IT	Italy
LA	Latin America
NL	Netherlands
NO	Norway
PL	Poland
PO	Portugal
SF	Swiss French
SL	Slavic
SG	Swiss German
SP	Spain
SU	Finland

Keyboard Code	Country
SV	Sweden
UK	United Kingdom
US	United States
YU	Yugoslavia

If you are using a secondary keyboard template, place the corresponding KEYB command in the AUTOEXEC.BAT file to ensure that the keyboard you want is in effect whenever your system starts.

DOS versions prior to 3.3 provide individual country keyboard commands in the form KEYB*xx*.COM, where *xx* is the two-letter code that represents a specific country.

KEYB supports the following exit status values:

Exit Status	Meaning
0	Keyboard successfully loaded
1	Invalid keyboard, code page, or syntax
2	Invalid keyboard definition file
4	CON device error
5	Error preparing code page

Examples

Use this command if you want DOS to use the United Kingdom keyboard template:

```
C:\> KEYB UK <ENTER>
```

This command selects the French keyboard template and informs DOS that the file KEYBOARD.SYS resides in the subdirectory C:\DOS:

```
C:\> KEYB FR,850,C:\DOS\KEYBOARD.SYS <ENTER>
```

Messages

Active code page not available from CON device

Make sure that you have placed the DISPLAY.SYS entry in CONFIG.SYS and issued the correct MODE command for the code page you want.

Bad command or file name

Either the KEYB*xx* command you are trying to execute does not reside on the current disk, or you are using DOS 3.3 or later, which uses the KEYB command rather than your KEYB*xx* command.

Bad or missing keyboard definition file

KEYB cannot locate the file KEYBOARD.SYS. Make sure that KEYBOARD.SYS resides in the current directory, or that you specify a complete path name to the file in the KEYB command line.

Code page requested (*nnn*) is not valid for given keyboard code

Your keyboard template and the code page you want are inconsistent. Specify a character code page whose character set corresponds to the keyboard template that you want.

Code page specified has not been prepared

Make sure that you have specified a correct DISPLAY.SYS entry in CONFIG.SYS and that you issue the corresponding MODE command (see MODE).

Current CON code page: *nnn*

KEYB is simply telling you the active code page (character set) for the CON device.

Current Keyboard code: *nnn* code page: *nnn*

KEYB is simply displaying the current keyboard template and code page for the CON device.

Invalid code page specified

The three-digit code page you have specified in the command line is invalid. Double-check the entry number of the code page you want and reissue the command.

Invalid keyboard code specified

The two-letter keyboard code you specified in the command line is invalid. Double-check the keyboard code you want and reissue the command.

KEYB has not been installed

KEYB is telling you that due to an error (whose error message should also be displayed), the KEYB command failed and a keyboard template was not installed.

One or more CON code pages invalid for given keyboard code

One of the three-digit code pages you specified is invalid for the keyboard code specified. Review the code page and keyboard code identifiers and reissue the command with the correct character set (code page).

Unable to create KEYB table in resident memory

Your system has insufficient RAM to install the memory-resident portion of the KEYB command. If you have installed other memory-resident software or a RAM drive, try restarting DOS without them.

LABEL

Specifies a disk volume label

DOS allows you to define a name (called a *volume label*) for each disk. Disk labels help you organize your floppy disks and also can prevent an inadvertent format of your hard disk.

Format

```
LABEL[target_drive:][volume_label]
```

target_drive: The disk drive that contains the disk to be labeled.

volume_label An 11-character volume label. All characters valid in DOS filenames are valid in volume labels.

Notes

Whenever you issue the DIR command, DOS displays the volume label of the disk for which it is displaying the directory, as shown here:

```
Volume in drive A is DOSLABEL
Directory of  A:\
```

It is also possible to use software to obtain the disk volume label from within your DOS programs. By so doing, you can ensure that the user has the correct disk in the drive.

If you do not specify a volume label in the command line, LABEL prompts you for one, as follows:

```
Volume in drive C is DOSDISK
Volume label (11 characters, ENTER for none)?
```

If you do not want to change the disk label, press ENTER; otherwise, type in the volume name you want.

Using DOS 4 or later, the FORMAT command prompts you to enter the disk volume label before it formats your hard disk. By assigning a unique label to the disk, you reduce the possibility of inadvertently formatting your hard disk.

The VOL command also displays volume label (see VOL).

Examples

The following command gives the name DOSDISK to the floppy disk in drive A:

```
C:\> LABEL A:DOSDISK <ENTER>
```

Since the command line contains a label name, LABEL does not prompt the user for information.

In the following command, the label name is not specified in the command line:

```
C:\> LABEL <ENTER>
```

In such a case, LABEL prompts:

```
Volume in drive C is DOSDISK
Volume label (11 characters, ENTER for none)?
```

If you are using DOS 5 or later, the LABEL command also displays the disk's serial number, as shown here:

```
Volume in drive C is DOSDISK
Volume Serial Number is 1234-5678
Volume label (11 characters, ENTER for none)?
```

Either enter the label name you want, or press the ENTER key to leave the current label name unchanged. After you press ENTER, if you are using DOS 5 or later, you will see another prompt that allows you to indicate whether you wish to delete the current volume label.

Messages

Cannot LABEL a NETWORK drive

The disk drive letter you have specified in the LABEL command corresponds to a network disk drive. DOS will not allow you to assign a label to a drive in the network.

Delete current volume label (Y/N)?

You have pressed ENTER in response to the prompt to enter a new volume label, and LABEL wants to know whether you want to delete the current volume label. If so, simply type **Y** and press ENTER; otherwise, type **N**.

Invalid characters in volume label

One or more of the letters you are using in the disk volume name is illegal.

No room in root directory

LABEL could not create the volume label because the root directory is full.

Volume in drive *n* is *volname*

LABEL is informing you of the current name for the disk in the drive specified.

Volume label (11 characters, ENTER for none)?

LABEL is prompting you to enter the name you want to assign to the disk.

LOADFIX

Loads a program into the conventional memory above 64Kb

In DOS 5 or later, you can use the DOS=HIGH,UMB CONFIG.SYS entry to load DOS into the high-memory area and to maintain a link to reserved memory for use by DEVICEHIGH and LOADHIGH to load device drivers and memory-resident programs into reserved memory. Each of these techniques frees conventional memory for DOS, possibly allowing commands to execute in the conventional address space below 64Kb.

Some programs cannot execute below 64Kb and will display the error message "Packed file corrupt" when you execute them. Using LOADFIX, you can direct DOS to load these programs above 64Kb, thus solving the problem.

Format

```
LOADFIX pathname [parameters]
```

pathname The path name of the program to execute.

parameters The program's optional command-line parameters.

Notes

LOADFIX requires DOS 5 or later. You will not need to use LOADFIX unless you experience the "Packed file corrupt" error message.

Example

The following command loads the program SOMEPROG.EXE above 64Kb in conventional memory:

```
C:\> LOADFIX  SOMEPROG.EXE  [parameters] <ENTER>
```

LOADHIGH

Loads a program into reserved memory

Conventional memory is your computer's memory from 0 to 640Kb. *Reserved memory* is the memory from 640Kb to 1Mb and normally is used for device I/O. *Extended memory* is the memory beyond 1Mb. Most programs run in conventional memory. Beginning with DOS 5, the LOADHIGH command lets DOS load memory-resident programs into the unused portions of reserved memory. By loading programs into reserved memory, more conventional memory is available to your applications programs.

Format

```
LOADHIGH pathname [parameters]
```

or

```
LH pathname [parameters]
```

pathname The complete path name, including the disk drive letter, of the program DOS is to load into reserved memory.

parameters The optional command-line parameters.

Notes

To use LOADHIGH, you must first load EMM386.EXE and direct DOS to save the link between reserved and conventional memory by using CONFIG.SYS's DOS=UMB entry as shown here:

```
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=UMB
```

If the reserved memory available is insufficient to load the program, DOS will load the program into conventional memory. DOS does not indicate when this has occurred.

DOS lets you abbreviate LOADHIGH as simply LH.

Example

The following command loads a memory-resident program named SOMEPROG.EXE into reserved memory:

```
C:\> LOADHIGH SOMEPROG.EXE <ENTER>
```

Message

Required parameter missing

The command line does not contain the name of a program to load into reserved memory. Invoke LOADHIGH again, specifying a program name.

MEM

Provides information on current memory usage

MEM is a DOS 4 or later command that displays your system's current memory usage. MEM is used most often by programmers.

Format

```
MEM[/DEBUG][/PROGRAM][/CLASSIFY][/FREE][/PAGE][/MODULE Name]
```

/DEBUG Tells MEM to display the location of each device driver in memory as well as each program's memory usage. This parameter can also be abbreviated as /D.

/PROGRAM Tells MEM to display all programs that currently reside in memory and a summary of their usage. This parameter can also be abbreviated as /P.

/CLASSIFY Classifies programs by memory use (DOS 5 or later).

/FREE A DOS 6 switch that directs MEM to display the free areas in conventional and upper memory. You can abbreviate this switch as /F.

/PAGE A DOS 6 switch that directs MEM to pause after each screenful of information. You can abbreviate this switch as /P.

/MODULE Name Directs MEM to display information about how a specific program is currently using memory.

Notes

With the introduction of the DOS 5 memory-management tools, the MEM command has become a very convenient way to determine your system's memory utilization. MEM will display information about

- Conventional, extended, and expanded memory use
- Available conventional and upper memory
- Program and device driver memory use

Examples

If you invoke MEM with no switches, MEM displays a summary of your memory use:

```
C:\> MEM <ENTER>

Memory Type        Total =  Used  +  Free
----------------  ------   ------   ------
Conventional        640K      57K     583K
Upper                59K      43K      16K
Adapter RAM/ROM     325K     325K       0K
Extended (XMS)     7168K    2624K    4544K
Expanded (EMS)     7488K    2704K    4784K
----------------  ------   ------   ------
Total memory      15680K    5753K    9927K

Total under 1 MB    699K     100K     599K

Largest executable program size       583K  (596528 bytes)
Largest free upper memory block        11K   (10784 bytes)
MS-DOS is resident in the high memory area.
```

The following command displays specifics about how programs and drivers are using memory:

```
C:\> MEM /DEBUG <ENTER>
```

If you are using DOS 6, you can display the previous output a screenful at a time using the /PAGE switch:

```
C:\> MEM /DEBUG /PAGE <ENTER>
```

Finally, the following MEM command tells you the memory use for a specific program, which in this case is COMMAND.COM:

```
C:\> MEM /MODULE COMMAND <ENTER>

COMMAND is using the following memory:
  Segment  Region        Size        Type
  -------  ------  -----------------  -------
   00667               2640   (3K)  Program
   00711               1040   (1K)  Environment
                   -----------------
  Total Size:          3680   (4K)
```

Message

xxxx is not currently in memory

The MEM command line included a /MODULE directive for a program that is not currently in memory. Make sure you are not specifying a file extension, but rather, only a program name.

MEMMAKER

Optimizes memory use on a 386 or higher

MEMMAKER optimizes your system's memory use by moving device drivers and memory-resident programs from conventional to upper memory. MEMMAKER may change your CONFIG.SYS, AUTOEXEC.BAT, and Windows SYSTEM.INI files.

Format

```
MEMMAKER [/BATCH] [/SWAP:Drive] [/UNDO]
       [/W:WindowsBuffer1,WindowsBuffer2]
```

/BATCH Directs MEMMAKER to select its default options for all prompts, running without user intervention.

/SWAP:Drive Specifies the drive letter of your original startup disk. The switch is only necessary if you are using a third-party software program that changes the startup drive to possibly a compressed drive.

/UNDO Directs MEMMAKER to undo its most recent changes to CONFIG.SYS, AUTOEXEC.BAT, and SYSTEM.INI.

/W:WindowsBuffer1,WindowsBuffer2 Specifies the amount of memory in Kb (1024 bytes) DOS reserves for Windows translation buffers. The default is 12Kb. If you don't use Windows, use the value 0,0.

Notes

If you run MEMMAKER in batch mode, MEMMAKER records information about each of its changes to the file MEMMAKER.STS. Using EDIT or TYPE, you can display the file's contents.

If MEMMAKER changes your CONFIG.SYS, AUTOEXEC.BAT, or SYSTEM.INI files, MEMMAKER stores each file's original contents in files with the OLD extension (such as CONFIG.OLD).

When Windows runs in protected mode, Windows allocates two buffers in the upper-memory area that it uses for input and output operations to devices that must operate beneath 1Mb. If you don't use Windows, use the value 0 for each buffer to prevent the buffer allocation.

MEMMAKER requires DOS 6.

Examples

The following command invokes MEMMAKER, allowing you to respond to MEMMAKER's prompts so you can control the changes made:

```
C:\> MEMMAKER <ENTER>
```

The following command directs MEMMAKER to prevent the allocation of the two 12Kb Windows translation buffers:

```
C:\> MEMMAKER /W:0,0 <ENTER>
```

MIRROR

Stores critical disk information

A DOS 5 command, MIRROR saves critical disk information to a file named MIRROR.FIL, from which the UNFORMAT command can later rebuild an inadvertently formatted or recovered disk. (If you recover a disk that is not damaged, you may scramble it beyond recognition.)

Format

```
MIRROR [d:][/Tdrive[-entries]]
  [[/Tdrive[entries]]...][/1]
MIRROR [d:][/PARTN]
MIRROR [d:][/U]
```

d: Specifies the drive letter for which you want MIRROR to record the critical disk information. If you don't specify a drive letter, MIRROR uses the default.

/1 Keeps only the latest information about the disk. By default, MIRROR makes a backup copy of the previous information, storing the information in the file MIRROR.BAK.

/Tdrive[–entries] Installs a memory-resident program that records information about every file deleted from the disk specified. The entries option specifies the number of deleted files MIRROR should track, from 1 to 999. If this information is available, the UNDELETE program uses it.

/U Tells MIRROR to remove the memory-resident delete-tracking program from memory, disabling file-delete tracking.

/PARTN
Saves the hard disk partition information to a floppy disk.

Notes

The MIRROR command saves a copy of the root directory and file allocation table. The more often you invoke MIRROR, the less likely you are to inadvertently lose files.

MIRROR stores information in a file named MIRROR.FIL, which resides in the root directory. The UNFORMAT command uses this information to rebuild a damaged disk. The size of MIRROR.FIL depends on your disk type.

If you invoke MIRROR with /Tdrive, MIRROR will create a file to store information about each file you delete. The UNDELETE command uses (but does not require) this file. If you want to track multiple drives, they must all be entered at the same time. To change an existing configuration, use the /U parameter to first disable tracking.

Use /PARTN to save a copy of the disk's partition table information to a floppy disk. Store the floppy disk safely.

Examples

The following command tells MIRROR to record the file allocation table and root directory for drive C and to install file delete-tracking for that drive:

```
C:\> MIRROR C: /TC <ENTER>
```

The next command tells MIRROR to save drive C's partition table information:

```
C:\> MIRROR C: /PARTN <ENTER>
```

MIRROR will prompt you for the letter of the disk drive containing the disk on which you want the partition table information written.

Messages

Cannot install. Already resident, or unknown status

You probably are attempting to install MIRROR for a second time. MIRROR can be invoked only once per user session. To use MIRROR with a different configuration, you must restart your system or use the /U parameter.

Drive *n* could not be processed

The drive letter you have specified is invalid. Make sure that you specify a valid drive letter, followed by a colon.

Error reading system area

MIRROR experienced a disk-read error while trying to read the disk's file allocation table or root directory. Disk-read errors are a sign of serious disk problems. If this error message occurs, perform a system backup immediately.

Insufficient memory to read all system info

Your system does not have enough RAM available for MIRROR to continue. If this error message occurs, remove memory-resident programs. You may want to modify your CONFIG.SYS entries also. Then reboot DOS.

Insufficient space for Mirror image file

There is not enough space available on the disk for MIRROR to store copies of the root directory and file allocation table. If possible, delete unnecessary files or move files to a new disk and invoke MIRROR again.

Mirror cannot operate with a network

The disk drive specified is a network drive, which MIRROR does not support.

There are no entries available in the root directory of the hard drive

MIRROR cannot create the file MIRROR.FIL in the root directory because the root directory is full.

Warning: Unrecognized DOS INT 25h/26h handler. Some other TSR programs may behave erratically while Delete-Tracking is resident! Try installing Mirror BEFORE your other TSR's.

MIRROR is warning you that it does not recognize the software DOS is using to read and write to disk. If you have installed another memory-resident program that intercepts disk operations (such as FASTOPEN), the program may not be compatible with MIRROR. Because the two programs are both intercepting the information DOS is reading from or writing to disk, a disk error that renders the disk unusable may occur. If this error occurs, determine which program is conflicting with MIRROR, and invoke the MIRROR program first.

MKDIR

Creates the subdirectory specified

DOS subdirectories let you organize your files by grouping related files into a list called a *directory.* The MKDIR command creates directories on your disk.

Format

```
MKDIR[drive:]path
```

or

```
MD[drive:]path
```

drive: Specifies the drive on which to create the subdirectory. If a drive is not specified, MKDIR uses the default drive.

path Specifies the name of the directory that MKDIR is to create.

Notes

The abbreviated form of the MKDIR command is MD.

Every DOS disk has a root directory (\) from which all other subdirectories grow. If you do not use DOS subdirectories, your disks are restricted to a limited number of files in the root directory.

You use MKDIR to create subdirectories. Whenever you create a subdirectory, one of two things happens. If the directory name starts with a backslash (as in \SUBDIR), DOS starts with the root directory to create the subdirectory. If the name does not start with a backslash (as in SUBDIR), DOS creates the directory within the current directory.

Follow these "rules of thumb" when you create DOS directories:

- DOS directory names conform to the same format as DOS filenames.
- DOS can process a *path name* (the representation of all directories and subdirectories) of up to 63 characters.

- If you do not specify a complete path name when you create a subdirectory, DOS assumes that you are creating the directory in the current directory.
- To manipulate directories on other disks, precede the directory name with a disk drive identifier (as in B:\CARS).
- Do not create directory names identical to the names of files contained in the same directory.
- Root directories on each disk are restricted to a specific number of files because of the disk layout. Subdirectories, however, can contain an unlimited number of files.

Examples

In the following example, MKDIR creates a directory called IBM in the root:

```
C:\> MKDIR \IBM <ENTER>
```

Similarly, the following command creates a subdirectory called NOTES in the IBM directory:

```
C:\> MKDIR \IBM\NOTES <ENTER>
```

Assuming that the current directory is still the root, this command:

```
C:\> MKDIR MISC <ENTER>
```

also creates a subdirectory in the root. In this case, the subdirectory name does not contain a slash, so MKDIR creates the directory in the current directory (which, in this case, is still the root).

Message

Unable to create directory

One of the subdirectories in the directory path may be invalid, a subdirectory or file with the same name as the new directory already exists on disk, or the root directory of the target disk drive is full.

MODE

Specifies device characteristics

The MODE command allows you to set parameters or characteristics for many types of devices. MODE specifies how DOS should interact with the device.

Format

To configure the video display:

```
MODE n
MODE [n],m,[t]
```

or, using DOS 5 or later:

```
MODE CON[:][COLS=columns][LINES=lines]
```

To configure COM ports:

```
MODE COM#[:] baud[,parity[,data[,stop[,P]]]]
```

or, using DOS 5 or later:

```
MODE COM#[:] BAUD=baud [PARITY=parity]
  [DATA=data][STOP=stop]
  [RETRY=r]
```

To configure a parallel printer:

```
MODE LPT#[:][cpl][,vli][,P]
```

or, using DOS 5 or later:

```
MODE LPT#:[COLS=cpl][LINES=vli][RETRY=r]
```

To accomplish printer redirection:

```
MODE LPT#[:]=COM#[:]
```

To set up international code pages:

```
MODE device CODEPAGE operation
```

To fine-tune your keyboard response, using DOS 5 or later:

```
MODE CON[:][RATE=rate][DELAY=delay]
```

n Specifies the screen-display attribute, which must be one of the following:

Value	Setting
40	Specifies 40-column display
80	Specifies 80-column display
BW40	Specifies a black-and-white 40-column display
BW80	Specifies a black-and-white 80-column display
CO40	Specifies a color 40-column display
CO80	Specifies a color 80-column display
MONO	Specifies a monochrome display

m Specifies the direction to shift the screen display, one character either to the left or to the right.

t Asks MODE to display a test pattern to aid in character alignment.

columns The number of characters per line on the screen: either 40 or 80.

lines The number of lines on the screen: 25, 43, or 50.

COM# Specifies the serial port number, such as COM1.

baud Specifies the device baud rate: 110, 150, 300, 600, 1200, 2400, 4800, 9600, or 19200. MODE requires only that you specify the first two digits of the baud rate.

parity Specifies the device parity: E for even parity, N for no parity, O for odd parity. The default is even parity.

data Specifies the number of data bits: 7 or 8. The default is 7 data bits.

stop Specifies the number of stop bits: 1 or 2. For 110 baud, the default is 2; otherwise, it is 1.

P Specifies continuous retries on time-out errors. (This parameter makes a part of MODE memory resident.)

r Specifies the retry action on time-out errors. (This parameter makes a part of MODE memory resident.) Options are as follows:

Value	Action
E	Returns a busy status error (the default)
B	Returns a busy status (same as using P parameter)
R	Returns a ready status
None	Performs no action

cpl Characters per line on the printer (80, 132).

vli Vertical lines per inch (6, 8).

LPT# Specifies the parallel printer number, such as LPT1.

device Specifies the device for which to manipulate a code page (CON, LPTn).

operation The code-page operation has several options. Use one of the following:

```
prepare=[[code_page][filename]]
select=code_page
refresh
/STATUS
```

rate Specifies the keyboard typematic rate in clock ticks. Values range from 1 to 32.

delay Specifies the length of time you must hold down a key before DOS begins to repeat the character. The values are 1, 2, 3, and 4; they correspond to 0.25, 0.50, 0.75, and 1.0 seconds.

Notes

MODE is one of the most difficult DOS commands, mainly because it is used for complex concepts (such as serial data communication or printer redirection). Unless you are using a serial laser printer or modem, chances are you will never have to issue the MODE command. If you are using the MODE command, your printer or modem documentation probably specifies the MODE command you need to issue. Place this command into the AUTOEXEC.BAT file so that you can forget about it.

By default, DOS uses the parallel printer port for printed data. If your printer is connected to a serial device, you can redirect the parallel output to serial device by using MODE.

If you are using DOS 3.3 or later, the MODE command allows preparation and selection of code pages for the screen display and printer. The PREPARE operation tells DOS to prepare a code page for use on the device specified. The SELECT operation chooses a specific code page to be used by the device. If you download a code page to your printer device while the printer is turned off, you can restore the code page by using the REFRESH operation. Finally, the /STATUS operation lists the code pages currently in use or prepared for the device specified.

MODE supports the following abbreviations:

CP	CODEPAGE
/STA	/STATUS
PREP	PREPARE
SEL	SELECT
REF	REFRESH

Examples

This command sets the screen display to 40 columns per line:

```
C:\> MODE 40 <ENTER>
```

If you are using DOS 5 or later, an equivalent command is

```
C:\> MODE CON: COLS=40 <ENTER>
```

The following command routes data from the parallel port (LPT1) to the serial port (COM1):

```
C:\> MODE LPT1:=COM1: <ENTER>
```

This command specifies the data communication parameters for COM1:

```
C:\> MODE COM1 96,N,8,1 <ENTER>
```

If you are using DOS 3.3 or later and an EGA monitor, this command prepares the display for the Multilingual and Portuguese code pages:

```
C:\> MODE CON CODEPAGE PREPARE=(850,860) C:\DOS\EGA.CPI <ENTER>
```

Note that you must specify the code-page information file (CPI) in the command line. In DOS 4 or later, this command line would be entered with an extra set of parentheses:

```
C:\> MODE CON CODEPAGE PREPARE=((850,860) C:\DOS\4201.CPI) <ENTER>
```

After you prepare a code page for a device, you select the code page by means of the SELECT option, as shown here:

```
C:\> MODE CON CODEPAGE SELECT=860 <ENTER>
```

If your printer supports code pages, the following command prepares the printer for the Multilingual and Portuguese code pages:

```
C:\> MODE LPT1 CODEPAGE PREPARE=(850,860) C:\DOS\4201.CPI <ENTER>
```

If you are using DOS 4 or later, this command line would be entered with an extra set of parentheses:

```
C:\> MODE LPT1 CODEPAGE PREPARE=((850,860) C:\DOS\4201.CPI) <ENTER>
```

Messages

Active code page for device *nnn* is *nnn*

You specified the CODEPAGE /STATUS option, which tells MODE to display the current code page for the device specified. This message is purely informational.

Code page not prepared

The MODE command includes either the CODEPAGE /STATUS option or CODEPAGE SELECT. If the command line contains /STATUS, this message is purely informational. If you issued a CODEPAGE SELECT command, you must first issue the CODEPAGE PREPARE option with the MODE command.

Codepage operation not supported on this device

Either the device name you have specified in the MODE command is invalid, or you have not installed the DISPLAY.SYS or PRINTER.SYS device driver in CONFIG.SYS.

Code pages cannot be prepared

The MODE command contains a CODEPAGE PREPARE option that cannot be completed. Make sure that you have installed the DISPLAY.SYS and PRINTER.SYS device drivers in the CONFIG.SYS file.

COM*n*: *baud, parity, data bits, stop bits, time out*

MODE is displaying the current data-communication parameters for the communication port specified. Use this message to determine your COM port settings.

COM port does not exist

The communications port specified in the MODE command line does not exist. Make sure that you are specifying COM1, COM2, COM3, or COM4 and reissue the MODE command.

Device error during operation

The MODE command could not complete successfully because of a variety of possible errors. Make sure that you have installed the DISPLAY.SYS or PRINTER.SYS device driver in CONFIG.SYS, that the device is turned on, and that you are using a valid code-page information file.

Device *nnnn* not prepared

CODEPAGE PREPARE failed. Make sure that you have specified the device name correctly and installed the device drivers DISPLAY.SYS and PRINTER.SYS in CONFIG.SYS.

Device or codepage missing from font file

The MODE command line contains a CODEPAGE PREPARE option that MODE cannot complete. Make sure that the command line specifies a valid code-page number and that the device name is correct.

Failure to access codepage font file

CODEPAGE PREPARE failed because MODE could not locate the code-page information file specified. Make sure that the code-page information file resides in the current directory or that a complete subdirectory path is specified.

Failure to access device

MODE could not access the device specified. Make sure that you have specified the device name correctly and installed any required device drivers in CONFIG.SYS.

Font file contents invalid

CODEPAGE PREPARE failed because MODE could not read the contents of the code-page information file. Reissue the command with the original DOS code-page information file.

Illegal device name

One of the device names specified in the MODE command is invalid. Make sure that you are specifying either LPT1 through LPT3 or COM1 through COM4; then reissue the MODE command.

Infinite retry of parallel printer timeout

You have specified the P parameter in the MODE command line to enable infinite retries if your printer has a timeout condition. When continuous retry is enabled, the computer will continually attempt to access the printer device (until the user presses CTRL-BREAK).

Invalid baud rate specified

Command line MODE COM*n* specifies an invalid baud rate.

LPT*n*: not redirected

MODE has set printer output back to specified parallel port.

LPT*n*: redirected to COM*n*:

MODE has redirected output from the parallel port LPT*n* to the serial communications port COM*n*.

LPT*n*: set for 80

MODE is simply telling you that it has set your printer to 80 characters per line. Depending on your printer type, MODE may not actually be able to set your printer as specified; regardless, MODE displays this message.

LPT*n*: set for 132

MODE is telling you that it has set your printer to 132 characters per line. Depending on your printer type, MODE may not actually be able to set your printer as specified; regardless, MODE displays this message.

No retry on parallel printer timeout

The MODE LPT*n* command line did not specify the P parameter, which tells DOS to perform continuous retries to access the printer device.

Previously prepared codepage replaced

MODE has prepared a new code page for the device specified, overwriting the previously prepared code pages. This message is purely informational.

Printer lines per inch set

MODE has set the number of lines of text per inch on your printer to either 6 or 8, as specified in the command line. Depending on your printer type, MODE may not be able to actually set your printer as specified in the command line; regardless, MODE displays this message.

MORE

Displays a command's output, a screenful at a time

MORE enables you to view a file or a program's output one screenful at a time. Press any key when you are ready to view the next screenful of information.

Format

```
DOS_command | MORE < DOS_file
MORE < DOS_file
```

Notes

The MORE command reads data from the standard input device, displaying the information on the standard output device, a page at a time, until an end-of-file marker is encountered. Each time a page of data is displayed on the screen, MORE displays the message:

```
-- MORE --
```

Press any key to continue the output or press CTRL-C to terminate the command.

Examples

The following command displays the files in the directory a screenful at a time:

```
C:\> DIR | MORE <ENTER>
```

By including the SORT command, you can enhance the previous command to display a sorted directory listing one screenful at a time:

```
C:\> DIR | SORT | MORE <ENTER>
```

To display the contents of a file named DATA.DAT one screenful at a time, use MORE as shown here:

```
C:\> MORE < DATA.DAT <ENTER>
```

MOVE

Moves a file from one directory to another

The MOVE command lets you move a file from one directory to another, or rename an existing directory. MOVE does not let you move directories.

Format

```
MOVE Source Target
```

Source Specifies the file or files that you want to move, or the directory you want to rename. MOVE fully supports wildcard characters.

Target Specifies the directory into which you want to move the files, or the desired name of a directory that you are renaming.

Notes

The MOVE command lets you move one or more files from one directory to another, or rename a directory. If a file exists in the target directory with the same name as you are using in the MOVE operation, the original file (if not read-only) is overwritten. MOVE lets you move files from one disk to another. If MOVE is successful, MOVE returns an exit status of 0.

The MOVE command lets you rename a directory. You cannot MOVE a directory, however. MOVE requires DOS 6.

Examples

The following MOVE command renames the directory OLDNAME to NEWNAME:

```
C:\> MOVE OLDNAME NEWNAME <ENTER>
```

This MOVE command moves the file BUDGET.DAT from the REPORTS directory into the directory FINANCE:

```
C:\> MOVE \REPORTS\BUDGET.DAT  \FINANCE <ENTER>
```

MOVE fully supports wildcard characters. The following command moves the files with the TAX extension from the BUDGET directory to the directory PAYMENTS:

```
C:\> MOVE \BUDGET\*.TAX  \PAYMENTS <ENTER>
```

You cannot use MOVE to move a directory. If you try to do so, the command will fail, as shown here:

```
C:\> MOVE \BUDGET \COMPANY\BUDGET <ENTER>
c:\budget => c:\company\budget [Permission denied]
```

Message

pathname => pathname [Permission denied]

You have tried to move a directory (MOVE lets you rename, not move directories) or the file specified exists as a read-only (or hidden, or system) file in the target directory.

MSAV

Examines your computer's memory and disks for viruses

The MSAV command scans your computer's memory and disk files for known viruses. If a virus is detected, MSAV may be able to remove the infection.

Format

```
MSAV  [[Drive:][...] | Path]
      [/A][/C | /S][/F][/L][/N][/P][/R]
      [/VideoSetting][/IM][/LE][/NGM][/PS2]
```

Drive: Specifies a disk drive whose files MSAV will scan for viruses. The ellipses (...) indicate you can specify several disk drives.

Path Specifies a directory whose files MSAV will scan for viruses.

/A Directs MSAV to scan all drives except A and B.

/C Directs MSAV to scan the specified files, removing viruses it encounters.

/F Directs MSAV to turn off the display of filenames it has scanned when you use /N or /P.

/L Directs MSAV to scan all drives except network drives.

/N Directs MSAV not to display information as it scans, but rather to write the information to the file MSAV.TXT, whose contents you can later view using the DOS TYPE command.

/P Directs MSAV to display a command-line interface as opposed to its graphical menu display.

/R Directs MSAV to create the file MSAV.RPT that lists the files scanned, number of viruses found, and the viruses removed. By default, MSAV does not create this file.

/S Directs MSAV to scan the specified disk for viruses, identifying but not correcting them.

/VideoSetting Directs MSAV to use a specific video mode. The value specified must be one of the following:

25	25-line text mode
28	28-line VGA text mode

43	43-line EGA and VGA text mode
50	50-line VGA mode
60	60-line mode for video 7 adapters
IN	Directs MSAV to use color
BW	Directs MSAV to use black and white
MONO	Directs MSAV to use monochrome
LCD	Directs MSAV to support an LCD monitor
FF	Directs MSAV to perform fast CGA screen updates
BF	Directs MSAV to use the BIOS for video output
NF	Directs MSAV to disable the use of alternative fonts
BT	Directs MSAV to support a graphics mouse cursor in Windows

/IM Directs MSAV not to support a mouse.

/LE Directs MSAV to exchange the left and right mouse select buttons.

/NGM Directs MSAV to use a text or block mouse cursor.

/PS2 Directs MSAV to support a PS/2 system mouse.

Notes

MSAV will scan your computer's memory and files on disk for known viruses. In most cases, MSAV can eliminate the virus. The more often you invoke MSAV, the less likely a VIRUS can damage your disk or files. Because MSAV executes very quickly, you might consider placing the MSAV command in your AUTOEXEC.BAT file, especially if you frequently exchange disks or files with other users.

MSAV requires DOS 6. See also VSAFE.

Examples

MSAV provides a set of menus that let you quickly select the options you desire. As such, most users simply invoke MSAV as follows:

```
C:\> MSAV <ENTER>
```

The following MSAV command directs MSAV to scan drive C for viruses, only reporting the existence of a virus without correcting it:

```
C:\> MSAV  C: /S <ENTER>
```

MSBACKUP

Invokes the DOS 6 menu-driven backup utility

MSBACKUP is the DOS 6 replacement for the BACKUP and RESTORE commands found in previous versions of DOS. MSBACKUP provides an easy way to back up and restore your disk files.

Format

```
MSBACKUP  [SetupFile] [/T[D|I|F]] [/BW|/LCD|/MDA]
```

SetupFile Specifies the name of a setup file that contains the desired backup options. If you don't specify a filename, MSBACKUP uses DEFAULT.SET.

/T Specifies the desired backup type: Differential, Incremental, or Full.

/BW Runs MSBACKUP in black and white.

/LCD Runs MSBACKUP using an LCD video display mode.

/MDA Runs MSBACKUP using a monochrome display adapter video mode.

Notes

MSBACKUP setup files specify which files you want to back up, backup options (such as verification), and the backup type. You create setup files within MSBACKUP. A differential backup operation backs up all files specified in the setup file that have changed since the last full disk backup. An incremental backup backs up all specified files that have been changed since the last incremental or full backup operation. A full backup operation backs up every file on your disk.

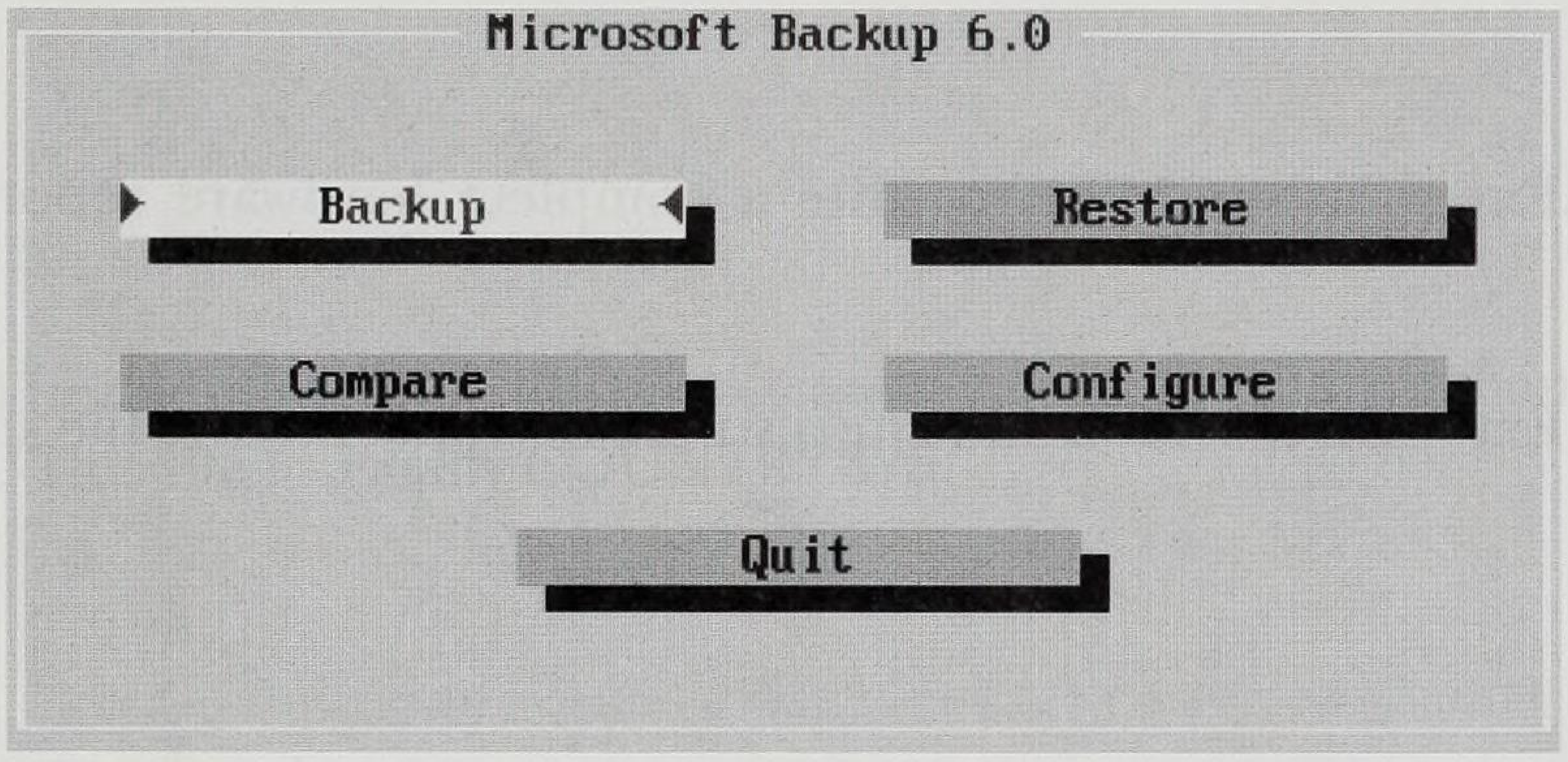
MSBACKUP requires DOS 6.

Example

MSBACKUP provides a menu-driven user interface. As such, most users will invoke MSBACKUP as follows:

```
C:\> MSBACKUP <ENTER>
```

MSBACKUP, in turn, will display its menu-driven interface as shown here:



Message

The DOS Backup is unable to find the file MSBACKUP.OVL
Press any key to continue...

MSBACKUP uses several overlay files (with the OVL extension). These files must reside in the current directory or in a directory specified in the command path.

MSD

Provides detailed technical specifics about your computer's hardware configuration

If you are using DOS 6 or Windows 3.1, you can use MSD to give you information about your computer's hardware and TSRs.

Format

```
MSD [/B] [/I] [/F Filename.Ext]
    [/P Filename.Ext] [/S Filename.Ext]
```

/B Directs MSD to run in black and white.

/I Directs MSD not to perform its initial hardware detection. Use this switch if MSD fails to start.

/F Filename.Ext Directs MSD to prompt you for your name, company, address and phone number and then writes them plus a complete hardware report to the file specified.

/P Filename.Ext Directs MSD to write a complete hardware report to the file specified.

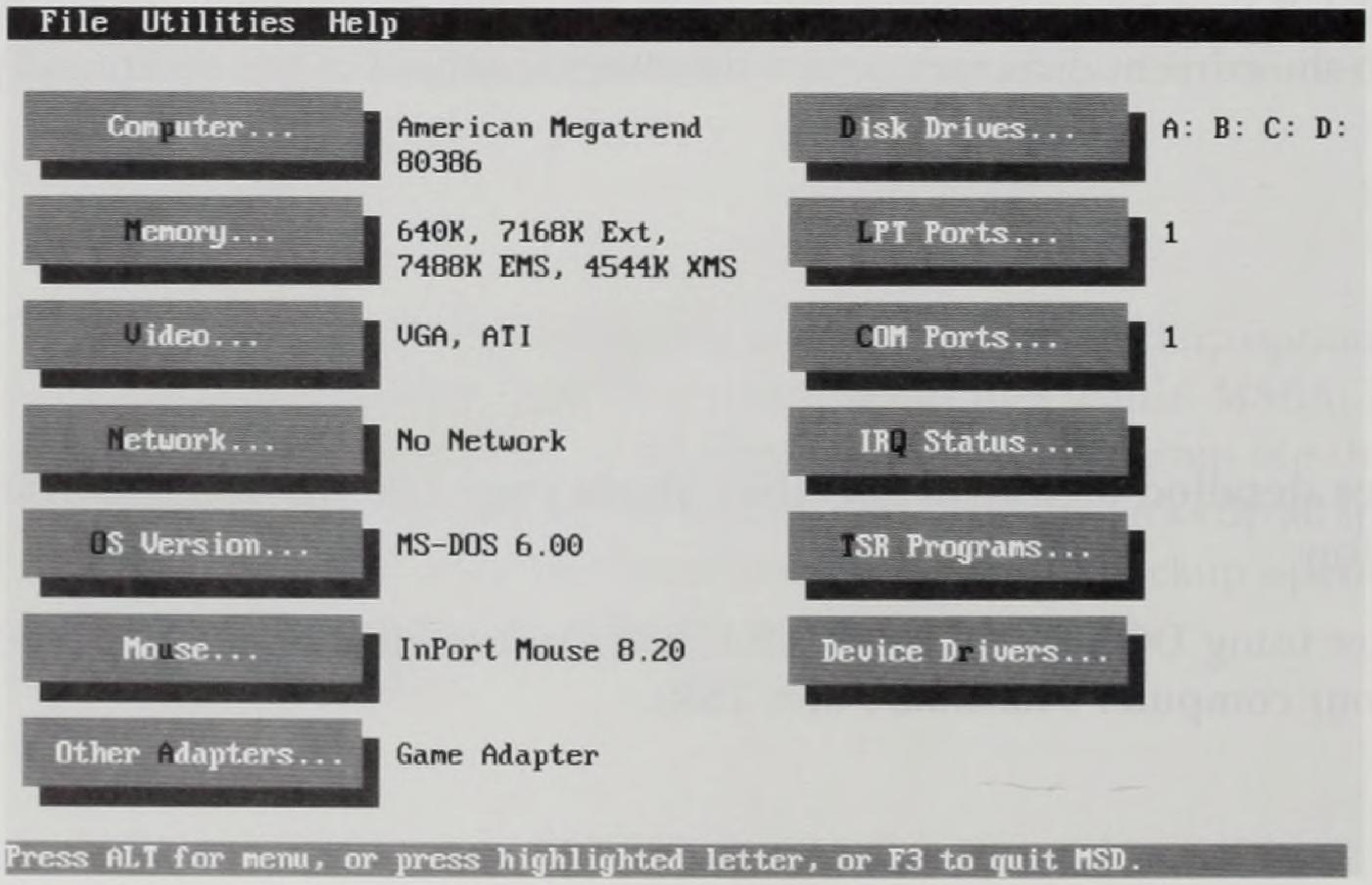
/S Filename.Ext Directs MSD to write a summary report to the file specified.

Notes

MSD is a Microsoft diagnostic utility that provides detailed information about your computer's memory, port, and IRQ settings, as well as device drivers and TSRs.

Example

When you invoke MSD, your screen displays a menu of settings options as shown here:



```
 File  Utilities  Help

 Computer...          American Megatrend        Disk Drives...      A: B: C: D:
                      80386
 Memory...            640K, 7168K Ext,          LPT Ports...        1
                      7488K EMS, 4544K XMS
 Video...             VGA, ATI                  COM Ports...        1

 Network...           No Network                IRQ Status...

 OS Version...        MS-DOS 6.00               TSR Programs...

 Mouse...             InPort Mouse 8.20         Device Drivers...

 Other Adapters...    Game Adapter

Press ALT for menu, or press highlighted letter, or F3 to quit MSD.
```

NLSFUNC

Provides device support for international code pages

NLSFUNC (National Language Support Function) is used by international DOS users to change character sets.

Format

```
NLSFUNC[file_spec]
```

file_spec The complete DOS file specification for the file that contains the country information. This file is usually COUNTRY.SYS.

Notes

The only users who will have to issue the NLSFUNC command are those concerned with code-page support for international character sets. If you are modifying code pages, you must issue the NLSFUNC command before you can invoke CHCP.

If you are using DOS 4 or later, you can use the CONFIG.SYS INSTALL= entry to load NLSFUNC during system startup.

Example

The following command informs DOS that the country information file (COUNTRY.SYS) resides in the directory on drive C:\DOS:

```
C:\> NLSFUNC C:\DOS\COUNTRY.SYS <ENTER>
```

Messages

File not found

NLSFUNC could not find the file COUNTRY.SYS. If COUNTRY.SYS is not in the current directory, specify the location of this file in the NLSFUNC command line.

NLSFUNC already installed

You have already invoked NLSFUNC. NLSFUNC installs memory-resident software that only requires you to invoke the command once per user session at the computer.

PATH

Defines a command-file search path

The PATH command defines the search path (the list of directories that DOS will follow) to locate an external command. By defining a command-file search path, you can easily execute your commonly used commands regardless of your current directory.

Format

```
PATH[drive:][path][;[drive:][path]...]
```

drive: The disk drive that DOS is to include in the command-file search path.

path A DOS subdirectory to be included in the command-file search path.

Notes

Each time you type in a command name from the DOS prompt, DOS first checks whether the command is an internal command that resides in RAM. If so, DOS executes the command. Each time DOS fails to locate a command internally, it searches the current directory or the directory specified in the command line. If DOS cannot find a matching EXE, COM, or BAT file, DOS checks whether you have defined a command-file search path. If so, DOS searches each of the directories specified in the command path for the command.

The PATH command enables you to define disk drives and subdirectories to be included in this search path.

To reduce overhead with each file reference, place the directories that are most likely to contain command files first in your PATH command. Each time DOS fails to locate a command in the current directory, it searches the first entry in your PATH path. If DOS still doesn't locate your file in that directory, it searches the second, third, and so on. If your commonly used commands reside in the directories specified at the end of the PATH command, DOS has to perform needless directory searches.

Examples

In this example, if DOS cannot find the command, it will search the root directories on drives C, B, and A, in that order:

```
C:\> PATH C:\;B:\;A:\ <ENTER>
```

In a similar manner, the following PATH command tells DOS to search C:\DOS, C:\BATCH, and then C:\MISC:

```
C:\> PATH C:\DOS;C:\BATCH;C:\MISC <ENTER>
```

Message

No Path

There is currently no command-file search path defined. Therefore, DOS will only search for your batch or executable files in the current or specified subdirectory.

PAUSE

Suspends execution of a batch file and displays an optional message

The PAUSE command temporarily suspends a batch file's processing until the user presses a key to continue.

Format

```
PAUSE[message]
```

message An optional message of up to 121 characters that PAUSE is to display when it suspends batch processing.

Notes

When it encounters a PAUSE command within a batch file, DOS will display a message similar to the following:

```
[optional message text]
Strike a key when ready . . .
```

The optional message is normally an instructional message the batch file displays to the user, such as to turn on the printer.

To continue batch processing, press any key; otherwise, press CTRL-BREAK. If you press CTRL-BREAK, DOS will display the following:

```
Terminate batch job (Y/N)?
```

To terminate the batch file, type **Y**; otherwise, type **N**.

The ECHO OFF command (see ECHO) suppresses the display of messages from PAUSE.

Examples

When DOS 5 or later encounters the PAUSE command with a batch procedure, as shown here:

```
PAUSE Insert a blank disk in drive B
```

it will pause and display

```
C:\> PAUSE Enter a blank disk in drive B
Press any key to continue . . .
```

In a similar manner, the command

```
PAUSE
```

will not display an instructional message but will still suspend processing. The following shows what happens if PAUSE is invoked with an earlier version than DOS 5:

```
C:\> PAUSE <ENTER>
Strike a key when ready . . .
```

Messages

Press any key to continue . . .

PAUSE is waiting for you to press a key to continue (DOS 5 or later).

Strike a key when ready . . .

PAUSE is waiting for you to press a key to continue (versions prior to DOS 5).

POWER

Uses the advanced power management (APM) specification supported by some computers to reduce the computer's power consumption when programs are idle.

Format

```
POWER [ADV[:MAX\REG\MIN]|STD|OFF]
```

ADV Specifies the type of power conservation desired: maximum, regular, or minimum.

STD Directs POWER to use only those power consumption capabilities supported by your computer or to turn off APM if the hardware does not support it.

OFF Turns off advanced power management.

Notes

Not all hardware supports advanced power management. Before you can use the POWER command, you must first install the device driver POWER.EXE within your CONFIG.SYS file. The format of the entry is the same as the POWER command with the exception that the driver supports the optional /LOW switch, which directs DOS to install the driver in low memory. By default, the driver uses upper memory if it is available.

Example

The following CONFIG.SYS entry installs the POWER.EXE device driver:

```
DEVICE=C:\DOS\POWER.EXE
```

After the driver is installed, you can use the POWER command to change its settings.

Message

Power Manager (POWER.EXE) is not installed

Before you can invoke the POWER command, you must install the device driver POWER.EXE within your CONFIG.SYS file.

PRINT

Prints a DOS file via the print queue

The PRINT command prints one or more ASCII files from the DOS prompt.

Format

```
PRINT [/D:device_name][/C][/T][/P]
  [/B:buffer_size][/M:max_ticks]
  [/Q:queue_size][/S:time_slice]
  [/U:busy_ticks]file_spec[...]
```

/D:device_name The name of the device that DOS is to use for the printer. The default device is PRN or LPT1.

/C Cancellation of printing the file that precedes the /C and all those that follow.

/T Cancellation of all the print jobs in the printer queue.

/P Addition of the preceding filename and all those that follow to the print queue.

/B:buffer_size The amount of memory (in bytes) that is set aside for PRINT. The default size is 512. By increasing this size in multiples of 512 (1024, 2048, 4096), you will improve PRINT's performance by decreasing the number of required disk I/O operations. However, increasing this value consumes memory.

/M:max_ticks The maximum number of CPU clock ticks that PRINT can consume before it must return control to DOS. This value can range from 1 to 255; the default value is 2. This parameter is only valid the first time that you invoke PRINT. Increasing this value will improve PRINT's performance, because it has more control of the CPU. However, if you make this value too large, the rest of your applications will become sluggish when PRINT is working.

/Q:queue_size The number of entries that the PRINT queue can store. This value can range from 4 to 32; the default value is 10. This parameter is only valid the first time you invoke PRINT.

/S:time_slice The number of CPU clock ticks that PRINT must wait before it can run again. (CPU clock ticks occur 18.2 times per second on the IBM PC.) This value must be in the range from 1 to 255; the default value is 8. This parameter is only valid the first time you invoke PRINT.

/U:busy_ticks The number of CPU clock ticks that PRINT will wait for before the printer becomes available for the next series of characters. This value can range from 1 to 255; the default value is 1. This parameter is only valid the first time you invoke PRINT.

file_spec The complete path name of the file to be added to, or removed from, the print queue. PRINT supports DOS wildcard characters.

Notes

The PRINT command sends files to the printer in a background mode, which means that you can continue processing in the foreground while printing takes place in the background.

The PRINT command lines the files in a print queue that are then printed in order of their entry in the queue.

Experiment with the PRINT parameters /B, /M, and /U. For starters, try

```
C:\> PRINT /D:LPT1: /B:4096 /M:24 /U:8 <ENTER>
```

Once you are satisfied with PRINT's performance and your overall system performance, place the associated PRINT command in the AUTOEXEC.BAT file, so you have a consistent PRINT installation for each user session.

Examples

The following command installs a print queue with storage for 32 files:

```
C:\> PRINT /Q:32 <ENTER>
```

Remember that many parameters are only valid the first time you issue a PRINT command.

The next command prints all the files in the current directory that have an extension of DAT:

```
C:\> PRINT *.DAT <ENTER>
```

In a similar manner, this command prints the CONFIG.SYS file:

```
C:\> PRINT CONFIG.SYS <ENTER>
```

The following command terminates all current print jobs:

```
C:\> PRINT /T <ENTER>
```

This command removes the AUTOEXEC.BAT file from the print queue:

```
C:\> PRINT AUTOEXEC.BAT /C <ENTER>
```

Messages

Access denied

The file specified is currently locked by another application. Wait a few minutes and repeat the PRINT command.

All files canceled by operator

You have canceled printing all of the files in the print queue via the /T command-line parameter.

Errors on list device indicate that it may be off-line. Please check it

Your printer may be turned off, or have been placed off line. Turn your printer on or place it on line and printing will continue.

File *FILENAME.EXT* canceled by operator

The file specified was removed via a /C parameter in the PRINT command line and will not be printed.

FILENAME.EXT File not found in print queue

PRINT could not find the file specified in a command that contains a /C parameter that tells PRINT to remove the file from the print queue. Redisplay the files in the print queue and then reissue the command.

FILENAME.EXT is currently being printed

PRINT is simply telling you the name of the file that it is currently printing.

FILENAME.EXT is in queue

PRINT is telling you that the file specified in the print queue is waiting to be printed.

FILENAME.EXT Pathname too long

The complete path name specified exceeds 63 characters. DOS will not process paths longer than 63 characters in length.

List output is not assigned to a device

The device name specified at PRINT's prompt for a target device is invalid. Reissue the command specifying a device name such as PRN, LPT1, or COM1.

Name of list device[PRN:]

PRINT is simply prompting you to enter the name of the target device that you want printed output to be written to. By default, PRINT uses the device PRN. If you simply press ENTER, PRINT will select this device. If your printer is attached to a different port, such as COM1, simply type in that device name.

No paper error writing device _nnnnn_

Your printer is out of paper. Simply take the printer off line, add additional paper, and place the printer back on line to resume output.

PRINT queue is empty

You have no files in the print queue, either printing or waiting to be printed.

PRINT queue is full

The printer queue is full and cannot hold an additional file until it prints another file. Remember, the /Q parameter sets a queue size of up to 32 entries. You can only specify /Q the first time that you invoke the PRINT command.

Resident part of PRINT installed

PRINT has installed the memory-resident software that DOS requires to handle your files and control of the printer.

PROMPT

Defines the system prompt that appears on the display

The PROMPT command defines how the DOS prompt appears on the screen display.

Format

```
PROMPT[prompt_string]
```

prompt_string The character string that defines the DOS prompt. It can contain characters or the following metastrings:

Character	Meaning
$$	$ character
$_	Carriage return/line feed
$b	| character
$d	Date
$e	ESC character
$g	> character
$h	Backspace
$n	Current drive
$p	Current drive and directory
$q	= character
$t	Current time
$v	DOS version
$l	< character

Notes

If no string is specified, PROMPT resets the system prompt to the default drive. The DOS prompt can be a help facility for users. Perhaps the most helpful is setting the prompt to display your current directory name, as follows:

```
C> PROMPT $p$g <ENTER>
C:\>
```

The PROMPT command creates an entry in the DOS environment. If you are using DOS 3.3 or higher, your batch files can use the named parameter %PROMPT% to determine and save the current system prompt.

Examples

The following command simply sets the DOS prompt to YES >:

```
C> PROMPT YES$g <ENTER>
YES>
```

The next command sets the system prompt to the current system time:

```
C> PROMPT $t <ENTER>
```

To display only the hours and minutes, you can change the prompt to

```
C> PROMPT $t$h$h$h$h$h$h <ENTER>
```

The following prompt, however, which is most common, simply displays the current drive and directory:

```
C> PROMPT $p$g <ENTER>

C:\>
```

RECOVER

Recovers a damaged disk or file

If a DOS disk or file becomes damaged and loses sectors (the areas on the disk that contain information), the RECOVER command can retrieve portions of the disk or file stored in good sectors. Data from the bad sectors cannot be recovered. DOS 6 does not support the RECOVER command.

Format

```
RECOVER[d:][path]file_name.ext
```

d: The drive identifier of the file or disk to be recovered. If you do not specify this parameter, RECOVER uses the default.

path The path name of the subdirectory that contains the file to be recovered. If you do not specify this parameter, DOS uses the default.

file_name.ext The name of the damaged file to be recovered.

Notes

The RECOVER command sometimes enables you to recover a portion of a file if the file develops a bad sector. However, RECOVER can only save a portion of the file from the undamaged sectors. Do not rely on this command; perform file and disk backups regularly. For a text file, you can edit the file later to restore the missing contents. For bad sectors in an executable file, do not execute the file—remember, the file is missing sectors.

If you use RECOVER to recover a complete disk, the command will create files in the root directory with names in the form FILE*nnnn*.REC, where *nnnn* is a four-digit number beginning with 0001 (FILE0001.REC).

RECOVER does not work with disk drives connected to a network.

Examples

The following command attempts to recover the contents of the disk in drive A:

```
C:\> RECOVER A: <ENTER>
```

In this case, RECOVER will create several files with names that are in the format FILE*nnnn*.REC, where *nnnn* represents the four-digit number of the file created. The command

```
C:\> RECOVER FILENAME.EXT <ENTER>
```

will simply recover the contents of the FILENAME.EXT file up to the damaged sector.

Messages

Cannot RECOVER a network drive

The disk drive you are specifying in the RECOVER command line is a network device. RECOVER does not work for a network disk.

The entire drive will be reconstructed,
directory structures will be destroyed.
Are you sure (Y/N)?

You have specified RECOVER to process the hard disk, and RECOVER is about to begin. Press the appropriate key to begin or cancel the process.

Press any key to begin recovery of the
file(s) on drive A:

You have specified RECOVER to process the floppy disk in drive A. Press any key to continue.

Warning - directory full. *n* file(s) recovered

The root directory of your disk drive is full. Copy the recovered files to a new disk, delete the current files, and repeat the RECOVER command to continue recovering files on the disk drive.

REM

Documents a DOS batch file

The REM command enables you to place comments or remarks within your DOS batch files to explain a batch file's processing.

Format

```
REM[message]
```

message A character string up to 123 characters long.

Notes

DOS does not execute lines containing the REM command; instead, DOS simply ignores the line and continues the batch file's execution with the next line. As a result, REM is a handy device for disabling a line that you might want to use later without having to retype the whole line; simply type **REM** at the front of the line, and DOS will ignore the line.

Place the ECHO OFF command at the start of your batch files to prevent DOS from displaying each REMark on the screen as a batch file executes.

At a minimum, your batch files should use REM to state the batch file's purpose, the author, and when the batch file was written.

```
@ECHO OFF
REM Name: DISPLAY.BAT
REM Purpose: Display the contents of each
REM          file specified.
REM Written by: Kris Jamsa
REM Date Written: 4/05/93
:LOOP
  IF '%1'=='' GOTO DONE
  FOR %%I IN (%1) DO TYPE %%I
  SHIFT
  GOTO LOOP
:DONE
```

Example

The following batch file uses REM to add several meaningful comments to the batch file presented in the "Notes" section that explain the batch file's processing:

```
@ECHO OFF
REM Name: DISPLAY.BAT
REM Purpose: Display the contents of each
REM          file specified.
REM Written by: Kris Jamsa
REM Date Written: 4/05/93
REM The format to invoke DISPLAY.BAT is:
REM
REM DISPLAY FILENAME.EXT[optional filenames]
REM
REM Loop through all of the filenames specified
:LOOP
  REM When %1 is null, the batch file is done
  IF '%1'=='' GOTO DONE
  REM By using the FOR command, the batch
  REM file supports wildcards
  FOR %%I IN (%1) DO TYPE %%I
  REM Assign the next filename to %1
  SHIFT
  GOTO LOOP
:DONE
```

RENAME

Renames the file(s) specified

The RENAME command renames one or more files on your disk.

Format

```
RENAME file_spec file_name
REN file_spec file_name
```

file_spec The complete path name of the file to be renamed. It can contain a drive and DOS subdirectory path. RENAME supports DOS wildcard characters for this parameter.

file_name The target filename for the rename operation. It cannot have a drive or DOS subdirectory path. The RENAME command supports DOS wildcard characters for this parameter.

Notes

Because of its frequency of use, DOS permits you to abbreviate RENAME as REN.

The REN command does not rename DOS subdirectories.

You must specify a target file after indicating the file you want to rename. The target and source files must reside in the same directory on the same disk drive, because RENAME does not copy the contents of a file. RENAME simply renames the file in its directory entry.

To move a file from one drive or directory to another, first copy the file to the desired drive or directory, then delete the file using DEL.

Examples

The following command renames the file CONFIG.BAK as CONFIG.SAV:

```
C:\> REN CONFIG.BAK CONFIG.SAV <ENTER>
```

The next command renames all the files in the current directory that have the extension BAK with the same filename and a new extension of SAV:

```
C:\> REN *.BAK *.SAV <ENTER>
```

Message

Invalid parameter

You have included a different drive or directory specification as part of your target filename. Check your spelling and target filename and enter valid parameters.

REPLACE

Replaces or updates selected files

The REPLACE command allows selective file replacements and updates when new versions of software become available.

Format

```
REPLACE source_file_spec
[target_file_spec][/A][/P][/R][/S][/U][/W]
```

source_file_spec The complete DOS source file specification for the files that REPLACE is to use. REPLACE supports wildcards.

target_file_spec The complete DOS destination file specification of the files being added or released.

/A Adds, rather than replaces, files to the target directory. With this parameter, REPLACE only places those files onto the target that are not currently present.

/P Displays a prompt to user before replacing or adding each file.

/R Replaces the files on the target location currently marked as read-only. Without this parameter, REPLACE stops replacement operations with the first file marked read-only.

/S Searches the subdirectories on the target location for other occurrences of the file to be replaced. This parameter cannot be used with /A.

/U Replaces only those files in the target directory that are older than the files in the source. This is a DOS 5 parameter.

/W Inserts a pause for a keypress before beginning to replace files, letting you change floppy disks.

Notes

The REPLACE command exists primarily to aid program developers when they have updates to programs that users may have distributed across several directories. Most users never issue the REPLACE command.

If you use REPLACE to update your fixed disk with a new version of DOS, keep in mind that REPLACE cannot update the hidden system files IO.SYS and MSDOS.SYS, which are required for your system to boot.

REPLACE uses the following exit status values:

Exit Status	Meaning
0	All files successfully replaced
2	Source files were not found
3	Source path was not found
5	File access violation
8	Insufficient memory
11	Invalid command-line syntax
15	Invalid drive specified

Examples

The following command replaces any files in the current directory that have the DAT extension using files that reside on drive A.

```
C:\> REPLACE A:*.DAT C: <ENTER>
```

The following command will replace each file on drive C that has the extension DAT with the corresponding file from drive A.

```
C:\> REPLACE A:*.DAT C:\ /S <ENTER>
```

Messages

Access denied '*pathname*'

REPLACE could not replace the file specified because it has been marked read-only. To replace a read-only file, use the /R parameter.

Add *pathname*? (Y/N)

You have invoked REPLACE with the /A and /P parameters to selectively add files to the target disk. In this case, to add the file to the disk, type **Y** and press ENTER. If you type **N** and press ENTER, REPLACE will not modify the file.

Adding *pathname*

The REPLACE command line contains the /A parameter, which tells REPLACE to add files found on the source disk to the files that reside on the target. In this case, REPLACE is simply telling you that it is adding the file specified to the target disk.

File cannot be copied onto itself '*path*'

A source file for the REPLACE command is identical to the target. Change the target filename and reissue the command or add a distinct disk drive or directory specifier.

n File(s) added

The REPLACE command line contains the /A parameter, which tells REPLACE to add files to the target disk, and REPLACE is simply telling you the number of files that REPLACE has added to the disk.

n File(s) replaced

REPLACE is simply telling you the number of files replaced on the target disk.

No files added

The REPLACE command line contains the /A parameter, which tells REPLACE to only add those files from the source disk that don't currently reside on the target disk. In this case, REPLACE found no files to add to the disk.

No files found '*path*'

No files existed in the source path specified. Make sure that the directory specified is valid and contains files, repeating the command as necessary.

No files replaced

REPLACE did not find any files on the target disk that needed to be replaced by files on the source.

Parameters not compatible

Two or more of the parameters specified in the REPLACE command line are incompatible. Review the format of the REPLACE command and reissue the REPLACE command.

Path not fou*nd* ' *\path* '

REPLACE could not locate the directory path specified. Make sure you are specifying a valid subdirectory, starting with the root directory if required.

Path too long

A subdirectory path name in the REPLACE command line exceeds the maximum of 63 characters.

Press any key to begin adding file(s)

The REPLACE command contains the /A parameter, which tells REPLACE to add files from the source disk to the target, and also /W, which tells REPLACE to pause, enabling you to insert the desired floppy disks into the correct disk drives.

Press any key to begin replacing file(s)

The REPLACE command line contains the /W parameter, which tells REPLACE to pause until the user presses a key. This enables you to insert the desired floppy disks into the correct disk drives.

Replace *FILENAME.EXT*? (Y/N)

The REPLACE command line contains the /P parameter, which enables you to perform selective file replacements. To replace the file specified, type Y and press ENTER. If you type N, REPLACE will not update the file specified.

Source path required

The REPLACE command line does not specify the location of the files to use in the replacement operation. Your command line must specify the files you want REPLACE to use.

RESTORE

Restores files saved by BACKUP

The BACKUP command places files onto a disk in a manner only accessible by RESTORE. To copy a file from the backup disk, you must use RESTORE. In DOS 6, if backup and restore operations are done by the MSBACKUP command.

Format

```
RESTORE source_drive:file_spec
  target_drive:file_spec[/P]
  [/S][/B:mm-dd-yy][/A:mm-dd-yy]
  [/E:hh:mm:ss][/L:hh:mm:ss][/M][/N][/D]
```

source_drive:file_spec The files to be restored. Each filename must match the name of the file as it was originally backed up; source_drive is the drive that contains the backup files.

target_drive:file_spec The disk drive to which the files will be restored.

/P Displays a prompt to the user before restoring those files that are read-only or have been modified since the backup.

/S Restores files contained in subdirectories.

/B:mm-dd-yy Restores only those files modified on or before the date specified.

/A:mm-dd-yy Restores only those files modified after the specified date.

/E:hh:mm:ss Restores only those files modified at or before the time specified. This parameter requires DOS 3.3 or later.

/L:hh:mm:ss Restores only those files modified at or after the time specified. This parameter requires DOS 3.3 or later.

/M Restores only those files modified since the last backup.

/N Restores only those files no longer existing on the target disk.

/D Displays the names of the files on the backup disk that match the file specification without actually performing the restore operation (DOS 5).

Notes

The RESTORE command will not restore the hidden system files that DOS uses to boot your system.

If you do not perform regular system backups, the RESTORE command is essentially worthless to you. Back up your disks regularly and place those backups in a safe location. DOS 3.3 enables you to keep a log of the files contained on your backup disks. Use this log to locate specific files on your disks for restoration. Remember to create meaningful labels for your backup disks.

RESTORE supports the following exit status values:

Exit Status	**Meaning**
0	Successful file restore
1	No matching files found to restore
3	User termination with CTRL-C
4	Fatal processing error; restore incomplete

Examples

The following command restores all files from the backup disk in drive A, including those in subdirectories:

```
C:\> RESTORE A: C:*.* /S <ENTER>
```

If the backup operation required several floppy disks, RESTORE will prompt you to place each subsequent backup disk in the specified drive as needed.

The next command restores all the files from the backup disk that contain the extension DAT:

```
C:\> RESTORE A: C:*.DAT /P <ENTER>
```

RESTORE will prompt you with

```
Warning! File FILENAME.EXT was changed after backed up.
Replace the file (Y/N)?
```

before it restores files that have been modified since the backup.

Messages

*** Files were backed up on *mm:dd:yy* ***

RESTORE is simply telling you the date that the files on the disk were backed up. If the date and time do not correspond to your last known BACKUP operation, you might not be using your latest disks.

*** Not able to restore file ***

The file specified may be currently in use by another program in the network. You should only perform BACKUP and RESTORE operations when your network is disabled.

*** Restoring files from drive *n* ***
Diskette *nnn*

RESTORE is simply telling you the disk drive letter and the disk number that it is currently using to restore files.

Insert backup diskette *n* in drive *n*
Strike any key when ready

RESTORE is ready for the next disk. Once you have inserted the disk in the drive specified, simply press a key, and RESTORE will continue its processing.

The last file was not restored disk full or file bad

RESTORE did not restore the last file displayed on the screen display because either the target disk is full or the file is bad. If your disk is full, remove unnecessary files from your disk before you retry restoring the files.

Restore file sequence error

RESTORE is performing a multiple-disk file restoration and you have inserted a disk out of order. Repeat the command, inserting the disks in the correct order.

Source and target drives are the same

The disk drives you have specified to restore from and to in the RESTORE command line are the same. Reissue the command with different source and target disk drives.

Source does not contain backup files

The disk in the source drive specified in the RESTORE command line does not contain files. Repeat the command, inserting a disk in the drive that contains files placed on the disk by the BACKUP command.

System files restored. Target diskette may not be bootable

RESTORE is warning you that it has restored DOS hidden system files that may render the disk unbootable. If your disk no longer boots, you may need to use the SYS command to make the disk bootable.

Target is full

The target disk cannot store additional files. The last file specified on the screen display may not have been copied to the disk.

Warning! Diskette is out of sequence
Replace diskette or continue if okay

You have inserted a disk in the wrong order. If you want RESTORE to continue with this disk, simply press ENTER. If you want to insert a different disk, do so and then press ENTER.

Warning! File *FILENAME.EXT*
changed after it was backed up.
Replace this file (Y/N)?

The RESTORE command line contains a /P parameter, which enables you to selectively restore files that have changed since the backup operation. If you want to restore the file specified, simply type Y and press ENTER. If you don't want RESTORE to restore the file, simply type N and press ENTER.

Warning! File *FILENAME.EXT*
is a read-only file.
Replace this file (Y/N)?

The RESTORE command line contains a /P parameter, which enables you to selectively restore files marked as read-only. To restore the file specified, simply type Y and press ENTER. If you don't want RESTORE to restore the file, simply type N and press ENTER.

Warning! No files were found to restore

RESTORE did not find any matching files to restore from the disk. Make sure you have correctly specified the files desired, including the subdirectories where they originally resided.

RMDIR

Removes the directory specified

The RMDIR command removes an empty directory from your disk.

Format

```
RMDIR[drive:]path
```

or

```
RD[drive:]path
```

drive: The drive from which the subdirectory will be removed. If this parameter is not specified, RMDIR uses the default drive.

path The name of the subdirectory to remove.

Notes

Because of its frequency of use, DOS permits you to abbreviate RMDIR as RD.

Use the RMDIR command to remove a specified directory. This command only removes empty subdirectories that do not contain files. RMDIR will not remove the current directory.

Examples

The following command attempts to remove the IBM subdirectory from the root directory of the current drive. If the directory contains files, RMDIR cannot remove the directory:

```
C:\> RMDIR \IBM <ENTER>
```

Similarly, the command

```
C:\> RMDIR \MISC\IBM\SALES <ENTER>
```

removes the empty subdirectory SALES from the directory \MISC\IBM on the current drive.

Message

Invalid path, not directory, or directory not empty

The directory name (or path name) you are specifying is invalid, the name specified is not a DOS subdirectory, or the directory still contains files or subdirectories. DOS does not remove a subdirectory that contains files.

SET

Places or displays DOS environment entries

When DOS boots, it reserves an area of memory called the *environment,* which provides a storage location for system specifics. DOS commands such as PROMPT and PATH place entries in the environment. The SET command sets or displays entries in the DOS environment.

Format

```
SET [name=[value]]
```

name The name of the DOS environment entry to which you are assigning a value.

value A character string that defines the assigned value.

Notes

The DOS environment always contains the COMSPEC= and PATH= entries. COMSPEC= tells DOS where to locate the command processor COMMAND.COM. The PATH= entry defines the current command search path.

Other commands that set environment entries include PROMPT, PATH, and APPEND. If you are using DOS 5, the DIRCMD entry defines the format DIR uses to display directory listings (see DIR). The SET command converts all entry names to uppercase.

The SET command, when entered with no parameters, displays the current environment.

You may have used the batch parameters %0 to %9. Beginning with DOS 3.3, you can use the named parameters that are environment entries whose names are surrounded by percent signs, such as %NAME%. Consider the following batch file:

```
CLS
TYPE %FILE%
```

When DOS encounters the %FILE% in the batch file, DOS will search your environment entries for an entry matching FILE=. Assuming that your environment contains the following:

```
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS;C:\BATCH
PROMPT=[$p]
FILE=TEST.BAT
```

DOS would display the contents of the file TEST.BAT.

Examples

With no command-line parameters, SET displays the current environment entries:

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
```

The following example creates a new environment entry called FILE and assigns it the value TEST.DAT:

```
C:\> SET FILE=TEST.DAT <ENTER>
```

You can verify this by again issuing the SET command:

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
```

```
PATH=C:\DOS
FILE=TEST.DAT
```

To remove the value for an entry, use SET as shown here:

```
C:\> SET FILE= <ENTER>
```

Invoking SET now displays

```
C:\> SET <ENTER>
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
```

Message

Out of environment space

The environment is full. Until you install memory-resident software such as PRINT, GRAPHICS, or a third-party product, DOS allows your environment to grow up to 32K as your processing needs require. However, if you install memory-resident software, your environment size becomes fixed. Try to invoke all of your SET commands prior to loading memory-resident software. Remember, by using SHELL= in CONFIG.SYS, you can specify the DOS environment size.

SETVER

Assigns a DOS version number to a program

Many older application programs, and in some cases older DOS commands, do not run in DOS 5. The SETVER command instructs DOS to respond that it is an earlier version when a specific program asks for the version number.

Format

```
SETVER[program_name][version]
SETVER program_name /DELETE [/QUIET]
```

program_name The complete path name of the program to which you want SETVER to assign a different DOS version.

version The DOS version number you want assigned to the program. The number must be in the range 2.11 to 9.99.

/DELETE The parameter to remove the version number assignment from the program.

/QUIET Suppresses SETVER messages for table entry delete operations.

Notes

Use the following CONFIG.SYS DEVICE= entry to load SETVER as a device driver:

```
DEVICE=C:\DOS\SETVER.EXE
```

SETVER creates a list of programs that you want run under different versions of DOS. Each time you change this list, you must restart your system for the change to take effect.

If you do not type in a program name, SETVER will display the current version number assignments.

SETVER supports the following exit status values:

Exit Status	Meaning
0	Successful assignment
1	Invalid command-line parameter
2	Invalid filename specified
3	Insufficient memory
4	Invalid version number format
5	Program name not found in list
6	MS-DOS system files not found
7	Invalid drive specified
8	Too many command-line parameters
9	Command-line parameters missing
10	Error reading MS-DOS system files
11	Corrupt version list
12	MS-DOS system files do not support version list
13	Insufficient space in list for a new entry
14	Error writing MS-DOS system files

Examples

The following command lists the current version number assignments:

```
C:\> SETVER <ENTER>
```

The next command tells SETVER to assign DOS 3.2 to the command EXE2BIN.EXE:

```
C:\> SETVER EXE2BIN.EXE 3.2 <ENTER>
```

To remove a command from the set version list, include the /DELETE parameter, as shown here:

```
C:\> SETVER TEST.EXE /DELETE <ENTER>
```

Messages

ERROR: Invalid version number, format must be 2.11 to 9.99

The version number you specified in the command line is not in the valid range of DOS version numbers from 2.11 to 9.99.

ERROR: Missing parameter

The SETVER command line does not contain both a filename and a version number to assign to the file. You must specify both.

Insufficient space in version table for new entry

The version table is full. If you want to add this entry to the table, you will have to delete an existing table entry. Invoke SETVER without any parameters to view the existing entries.

Version table successfully updated
The version change will take effect the next time you restart your system

SETVER is informing you that the command was successful and that you must restart your system for the change to take effect.

Warning! Lying about version numbers to a program may have bad effects. Microsoft will not be held responsible for this event

The SETVER command is assigning a strange DOS version number to a file. SETVER is warning you that although the program will run using the version number specified, the program might not run correctly. Make sure you entered the correct version number, and if not, repeat SETVER with a correct version number.

SHARE

Installs DOS file-sharing support

SHARE lets users in a network access the same file at the same time.

Format

```
SHARE [/F:file_space][/L:locks]
```

/F:file_space Memory allocation (in bytes) for the area in which DOS will store file-sharing information. Each open file requires 11 bytes plus the length of the filename (up to 63 characters). The default file space is 2048 bytes.

/L:locks The number of files that can be locked at the same time. The default number is 20.

Notes

You issue the SHARE command only if your computer is part of a local area network. Ask the person responsible for the network for the proper SHARE values.

DOS versions later than 3.0 support file and record locking. Each time a file is opened with file sharing installed, DOS checks whether the file is locked against the open operation. If it is, the file cannot be opened. In addition, DOS checks for locking during each read and write operation.

The SHARE command invokes a memory-resident program that performs the file and record locking. SHARE can only be invoked once. When it has been invoked, the only way to remove file sharing is to reboot. The SHARE command slows down the speed of all your file operations, so only install it when file sharing is in effect.

If you are using DOS 4 or later, you can use the CONFIG.SYS INSTALL= entry to install file sharing each time your system starts. If you are using DOS 4 and disk partitions larger than 32Mb, make sure you invoke SHARE to prevent damage from a program using file control blocks.

Examples

The following command invokes file sharing with default values of 2048 and 20 locks:

```
C:\> SHARE <ENTER>
```

The next command installs file-sharing support with 40 locks:

```
C:\> SHARE /L:40 <ENTER>
```

Message

Not enough memory

Your computer does not have enough memory to invoke the SHARE memory-resident software. If possible, remove other memory-resident software and RAM drives.

SHIFT

Shifts each batch parameter left one position

Your DOS batch files can easily access the first nine batch file command-line parameters using the symbols %1 through %9. If the user invokes your batch file with more than nine parameters, you can use the SHIFT command to access them.

Format

```
SHIFT
```

Notes

This command shifts each batch parameter one position to the left. If more than ten parameters are passed to a DOS batch procedure, you can use the SHIFT command to access each parameter past %9. If no parameter exists to the right of a parameter, SHIFT will assign the parameter a NULL string.

There is no way to undo the effects of the SHIFT command. A parameter in the leftmost position (%0) is lost once SHIFT has executed.

Example

The following batch file displays all batch parameters specified on the command line:

```
@ECHO OFF
:LOOP
  SHIFT
  IF '%0'== '' GOTO DONE
  ECHO %0
  GOTO LOOP
:DONE
```

If the previous file was named TEST.BAT, invoking the batch file by entering

```
C:\> TEST 1 2 3 4 <ENTER>
```

would display

```
C:\> TEST 1 2 3 4 <ENTER>
1
2
3
4
```

SMARTDRV

Improves system performance by installing a disk cache, which reduces disk operations

SMARTDRV is a memory-resident program that reduces the number of slow disk read and write operations by buffering information in a computer's fast electronic RAM.

Format

```
SMARTDRV [[Drive[+|-]...] [/E:ElementSize]
[InitialSize][WindowsSize]]
[/B:ReadAheadBuffer] [/C][/L][/Q][/R][/S]
```

Drive[+|–] A drive letter followed by a plus sign enables read and write caching for that drive. A minus sign disables caching. If you don't specify a plus or minus sign, SMARTDRV enables read caching while disabling write caching. The ellipses (...) indicates that you can specify several drives.

/E:ElementSize Specifies the amount of information (in bytes) SMARTDRV moves to or from disk at one time. Valid element sizes are 1024, 2048, 4096, and 8192. The default is 8192.

InitialSize Specifies in Kb (1024 bytes) the size of SMARTDRV's initial cache. In general, the larger the cache, the better your system performance.

WindowsSize Specifies the size in Kb (1024 bytes) to which SMARTDRV will reduce the cache size when Windows runs, to give Windows more memory with which it can operate.

/B:ReadAheadBuffer Specifies the size in Kb (1024 bytes) of a buffer into which SMARTDRV reads extra disk sectors in attempting to reduce subsequent disk read operations.

/C Directs SMARTDRV to immediately flush all cached (output) data to disk.

/L Loads SMARTDRV into conventional memory, as opposed to upper memory.

/Q Directs SMARTDRV not to display initialization error messages.

/R Directs SMARTDRV to clear the current cache contents and to restart.

/S Directs SMARTDRV to display status information.

Notes

Because your disk drive is a mechanical device, it is much slower than your computer's electronic components. SMARTDRV uses your computer's extended memory to create a large buffer into which it can read extra information in an attempt to satisfy future disk read operations. In addition, SMARTDRV buffers information programs write to disk, later performing the actual disk write operation at a more suitable time.

The SMARTDRV disk cache eliminates your need to use the DOS CONFIG.SYS BUFFERS entry. Depending on the amount of extended memory your system contains, the size of the SMARTDRV buffer you allocate will differ. Use the following table as a guideline:

Extended Memory	Initial Buffer	Windows Buffer
1 to 2Mb	1Mb	256Kb
3 to 4Mb	1 to 1.5Mb	512Kb
5 to 6Mb	2Mb	1Mb
>6Mb	2Mb	2Mb

Place the SMARTDRV command in your AUTOEXEC.BAT file.

Examples

The following SMARTDRV command allocates an initial 2Mb buffer for drive C, allowing Windows to reduce the buffer size to 1Mb:

```
SMARTDRV C+ 2048 1024
```

SMARTDRV allows you to specify the buffering of several disks. The following command selects a 1Mb buffer and enables caching for drives C and D:

```
SMARTDRV C+ D+ 1024
```

Using the /S switch, the following command displays SMARTDRV's status information:

```
C:\> SMARTDRV /S <ENTER>
Microsoft SMARTDrive Disk Cache version 4.0
Copyright 1991,1992 Microsoft Corp.

Room for 128 elements of 8,192 bytes each
There have been 689 cache hits
    and 208 cache misses

Cache size: 1,048,576 bytes
Cache size while running Windows: 262,144 bytes

            Disk Caching Status
drive   read cache    write cache   buffering
-------------------------------------------
  A:       yes           no           no
  B:       yes           no           no
  C:       yes           yes          no

For help, type "Smartdrv /?".
```

Because SMARTDRV may perform write caching, information you thought was recorded to disk may actually still reside in memory.

Before you restart or turn off your computer, you should use SMARTDRV's /C switch to first flush all the information to disk:

```
C:\> SMARTDRV /C <ENTER>
```

Messages

You cannot load SMARTDrive when a shell program (such as MS-DOS shell) or Windows is running...

You attempted to install the SMARTDRV disk-caching software while temporarily shelled to DOS. To install the caching software, first exit the current shell program.

SMARTDrive cannot be loaded because the XMS driver HIMEM.SYS is not loaded.

SMARTDRV creates its disk cache in extended (XMS) memory. Before DOS can use extended memory, you must install the HIMEM.SYS device driver within your CONFIG.SYS file.

SORT

Sorts and displays the contents of a file or redirected program output

The SORT command allows you to view a file's contents or a program's output in sorted order.

Format

```
DOS_COMMAND | SORT[/R][/+n]
SORT[/R][/+n] < file
```

/R Data sort in reverse order.

/+n Number of the column on which to sort the data.

Notes

The SORT command reads data from the standard input device, sorting the information and displaying it on the standard output device until an end-of-file marker is encountered. SORT is usually used with the < and | redirection operators.

SORT does not make any distinction between upper- and lowercase letters.

Examples

The following command displays a sorted directory listing:

```
C:\> DIR | SORT <ENTER>
```

The following command tells SORT to sort the information contained in the DATA.DAT file:

```
C:\> SORT < DATA.DAT <ENTER>
```

In a similar manner, the command

```
C:\> SORT /R < DATA.DAT <ENTER>
```

tells SORT to sort the same file, this time in reverse order.

If your data file contains

```
Bill M
Mary F
Kris M
Kal  M
Jane F
Mike M
Ed   M
```

then the following command

```
C:\> SORT /+6 < FILENAME.EXT <ENTER>
```

tells SORT to sort the file based on the data starting in column 6. In this case, SORT will display

```
Mary F
Jane F
Bill M
Kris M
Kal  M
Mike M
Ed   M
```

SUBST

Substitutes a drive name for a path name

Some older programs do not support DOS subdirectories. If you are using such a program, you can trick the program into using a subdirectory by substituting the desired directory name with a disk drive letter.

Format

```
SUBST [d:][path_name][/D]
```

d: The drive identifier that will be used to reference the path.

path_name The path name to be abbreviated.

/D The parameter to remove a previous disk substitution.

Notes

Because path names can become large, you can use the SUBST command to substitute a drive identifier for a path name. If you invoke SUBST without any parameters, current substitutions will be displayed. By default, the last disk drive that DOS will reference is drive E.

```
C:\> SUBST E: \DIRNAME <ENTER>
```

To specify a disk drive greater than E, use the LASTDRIVE= entry in CONFIG.SYS. For instance, the following entry tells DOS to support the drive letters A through K:

```
LASTDRIVE=K
```

Do *not* use the following commands with a substituted disk:

BACKUP
CHKDSK
DISKCOMP
DISKCOPY
FDISK
FORMAT
LABEL
RECOVER
RESTORE
SYS
UNDELETE
UNFORMAT

Examples

In the following example, DOS enables you to abbreviate the subdirectory named \REPORTS\1993 to the drive letter E:

```
C:\> SUBST E: \REPORTS\1993 <ENTER>
```

A command such as

```
C:\> DIR E: <ENTER>
```

will then display the contents of the directory. If that subdirectory contains other subdirectories, you can still use the drive letter, as shown here:

```
C:\> DIR E:SUBDIR <ENTER>
```

Invoking the SUBST command without command-line parameters displays current substitutions:

```
C:\> SUBST <ENTER>
E: => C:\REPORTS\1993
```

Messages

Cannot SUBST a network drive

The SUBST command failed because the disk drive specified in the command line refers to a network disk. You cannot substitute a network disk drive.

Path not found

SUBST could not locate the subdirectory specified in the command line to substitute.

SYS

Creates a bootable disk

To start your system, DOS requires that two hidden files reside in the disk's root directory. Use the SYS command to copy these hidden files from a bootable disk to a newly formatted disk.

Format

```
SYS source_drive: target_drive:
```

source_drive: A parameter available beginning with DOS 4 to specify from where the DOS system files should be copied.

target_drive: The target drive for the hidden operating system files.

Notes

The SYS command enables you to transfer to the target disk the hidden operating system files that perform the initial system startup. The SYS command does not copy the COMMAND.COM file to the target disk. To do so, you must use the COPY command.

DOS will only transfer files to an empty target disk or to a disk that was formatted previously with the /S or /B parameter (see FORMAT).

SYS does not work with a disk that uses the JOIN or SUBST command.

Prior to DOS 4, the hidden system files had to be the first two files in the root directory. If the root directory already contains files, you probably won't be able to transfer the hidden files to the disk. In such cases, SYS will display the following message:

```
No room for system on destination disk
```

Example

In the following example, SYS will transfer the hidden operating system files to the disk in drive A:

```
C:\> SYS A: <ENTER>
```

Messages

Cannot SYS to a network drive

The command failed because DOS will not copy system files to a network device.

Incompatible system size

SYS could not copy the hidden system files as contiguous files on the target disk. The disk may not be bootable.

Insert destination disk in drive *n*
and strike any key when ready

SYS is prompting you to insert the disk that you want SYS to transfer the system files to, into the disk drive specified.

Insert system disk in drive *n*
and strike any key when ready

SYS is prompting you to insert a DOS disk in the disk drive specified so that it can copy the hidden system files from the disk to the target disk specified in the SYS command line.

Invalid drive specification

The target disk drive in the SYS command line is invalid. Make sure you include a colon after the drive letter and you are using a valid disk drive.

No room for system on destination disk

SYS could not place the hidden system files on the target drive specified. The hidden DOS files must be the first files that DOS places on a disk. If SYS displays this message, you will very likely have to use FORMAT /S to create a bootable system disk. If you are using a fixed disk, make sure that you completely back up all of the files on your disk before doing so. Once the FORMAT /S command completes, you can restore the original files to the disk drive, adding the required DOS command files.

No system on default drive
Insert system disk in drive *n*
and press any key when ready

SYS could not find the hidden files on the disk in the current drive. Place a bootable DOS disk in the drive specified and press ENTER.

TIME

Displays and sets the system time

DOS uses the current system time to update the directory each time you create or modify a file.

Format

```
TIME[HH:MM[:SS[.hh]]][AM | PM]
```

HH:MM

The desired hours (0 to 23) and minutes (0 to 59).

SS

The desired seconds (0 to 59).

hh

The desired hundredths of a second (0 to 99).

Notes

If you are using DOS 5 or later, you can enter a time based on a 12-hour clock and include either the letters AM or PM following the time as appropriate. As such, the following TIME commands are identical:

```
TIME 14:30
TIME 2:30P
TIME 2:30PM
```

Examples

If you do not specify a time on the command line, as in

```
C:\> TIME <ENTER>
```

TIME will prompt you for one:

```
Current time is 16:08:41.15
Enter new time:
```

To leave the time unchanged, simply press ENTER. Otherwise, type in the time desired.

The following command sets the clock to 12:00 noon:

```
C:\> TIME 12:00 <ENTER>
```

The next command sets the clock to midnight:

```
C:\> TIME 00:00:00.000 <ENTER>
```

Message

Invalid time
Enter new time:

The time you have typed in is invalid. Enter a new time in the format hh:mm:ss.nnn and press ENTER. TIME only requires you to enter the hours and minutes.

TREE

Displays directory structure

DOS refers to the directories on your disk as the *directory tree*. The TREE command displays all of the directories on your disk and, optionally, the files each directory contains.

Format

`TREE[d:][path][/F][/A]`

d: The drive identifier of the disk for which TREE is to display the directory structure.

path The name of a directory at which you want the directory tree to begin. If you don't specify a directory, DOS uses the root directory.

/F The parameter that displays the name of each file in a directory as well.

/A The parameter that displays the directory tree using standard text characters instead of graphics characters. This parameter requires DOS 4 or higher.

Notes

Beginning with DOS 4, the TREE command uses graphics-like characters to display the directory structure.

If you use the I/O redirection operators, you can print the directory tree as well, as shown here:

```
C:\> TREE > PRN <ENTER>
```

Examples

The following command displays the directory structure of the disk:

```
C:\> TREE <ENTER>
```

To display the files in each directory, use the /F parameter:

```
C:\> TREE A:/F <ENTER>
```

Message

No subdirectories exist

The TREE command did not find any DOS subdirectories on the disk specified. This message is purely informational.

TYPE

Displays a text file's contents

The TYPE command displays the contents of a text file on the screen.

Format

```
TYPE file_spec
```

file_spec The complete DOS file specification for the file to be displayed. It can contain a drive identifier and path name.

Notes

The TYPE command is restricted to ASCII files. Do not use TYPE with files containing a COM or EXE extension, because these files contain characters that might cause the computer to beep and display meaningless characters on the screen.

When you use the TYPE command, don't forget that you can control the output by using the MORE command:

```
A> TYPE FILENAME.EXT | MORE
```

and also by pressing the CTRL-S key combination. When it encounters the CTRL-S, DOS suspends scrolling the current output until you press another key.

Examples

The following command tells TYPE to display the contents of the CONFIG.SYS file:

```
C:\> TYPE \CONFIG.SYS <ENTER>
```

In a similar manner, the command

```
C:\> TYPE \AUTOEXEC.BAT <ENTER>
```

tells TYPE to display the contents of the file AUTOEXEC.BAT that resides in the root directory.

UNDELETE

Recovers a previously deleted file

When you delete a file, the file actually remains on your disk until DOS overwrites the file's contents with another file. The DOS 5 UNDELETE command recovers recently deleted files. If you don't immediately recover a deleted file, the file may be overwritten by the next command, making its undeletion impossible.

Format

```
UNDELETE[pathname][/LIST][/DT | /ALL | /DOS]
```

or in DOS 6

```
UNDELETE [/LIST | /ALL | /PURGE[:Drive] | /STATUS
          | /LOAD | /U | /S[:Drive] | /TDrive[-entries]]
```

pathname The subdirectory path of the file or files you want to undelete. If you don't specify a path name, UNDELETE uses the current directory.

/LIST The parameter that lists all of the files available for recovery.

/DT The parameter for recovery of only those files listed in the delete tracking (DT) file.

/ALL The parameter for recovery of all of the files listed by DOS as deleted without a prompt to enter the first letter of the file. Instead, UNDELETE assigns the character # to the first letter of each file. If a file exists with the same name, UNDELETE will attempt instead to use the characters, % & - 0 1 2 3 4 5 6 7 8 9, in that order.

/DOS The parameter for recovery of only those files listed by DOS as deleted (ignoring the delete tracking file). DOS will prompt you to enter the first letter of each filename.

/LOAD Loads UNDELETE's memory-resident software using the files in the file UNDELETE.INI. If the file does not exist, UNDELETE uses default values. This is a DOS 6 switch.

/PURGE[:Drive] Deletes (purges) the contents of the SENTRY directory on the drive specified. If you don't specify a drive, UNDELETE uses the current drive. This is a DOS 6 switch.

/S[:Drive] Directs UNDELETE to initiate data sentry protection on the drive specified. If you don't specify a drive, UNDELETE uses the current drive. This is a DOS 6 switch.

/TDrive[–entries] Loads a memory-resident delete tracking program that tracks from 1 to 999 deleted files in the root directory file PCTRACKR.DEL. This is a DOS 6 switch.

Notes

If you are recovering files using /DOS, the recovered files may contain errors due to file fragmentation.

If you inadvertently delete a file, do not copy any files to the disk until you successfully recover the deleted file.

When DOS deletes a file, it replaces the first character in the file's name with a Greek symbol. To recover a file, UNDELETE may ask you to type in the file's original first character.

In DOS 6, UNDELETE incorporates many of the features of the DOS 5 MIRROR command, which no longer exists. The DOS 6 UNDELETE command provides two types of undelete protection. First, delete-file tracking (also in DOS 5) stores information about each file you delete, such as where the file began and the clusters it consumed. Using the delete-file tracking information is effective until the file's disk storage locations are overwritten. Second, the DOS 6 UNDELETE command supports sentry protection in which files are not actually deleted from disk, they are moved to the SENTRY directory. If disk space becomes low, UNDELETE will purge the SENTRY directory. You can also purge the files yourself using UNDELETE's /PURGE switch.

The DOS 6 UNDELETE command lets you specify various options using the root directory file UNDELETE.INI. Use UNDELETE's /U switch, to create the file.

Examples

The following UNDELETE command displays the names of all files available for recovery (using DOS) in the current directory:

```
C:\> UNDELETE /LIST /DOS <ENTER>
```

If you have enabled file-delete tracking using MIRROR, UNDELETE uses the information MIRROR has saved.

The following command undeletes all of the recoverable files in the current directory based on delete tracking information:

```
C:\> UNDELETE *.* /DT <ENTER>
```

Likewise, the following command undeletes all of the recoverable files in the current directory using only DOS file information:

```
C:\> UNDELETE *.* /DOS <ENTER>
```

The following DOS 6 command loads UNDELETE's data sentry protection:

```
C:\> UNDELETE /S <ENTER>
```

This DOS 6 command loads UNDELETE's file-delete tracking software, just as you would use the MIRROR command in DOS 5:

```
C:\> UNDELETE /TC <ENTER>
```

Messages

/DT and /DOS are mutually exclusive

Your command line contains the /DT and /DOS switches telling UNDELETE to access files marked as deleted by the delete tracking file and DOS. Use either one switch or neither.

All of the clusters for the file are available.
Do you want to recover this file? (Y/N)

UNDELETE is informing you that the file is fully recoverable. If you want to recover the file, type **Y**. If you don't want to recover the file, type **N**.

Cannot operate on a SUBST drive

The disk drive specified in the command line is a logical drive created by the SUBST command. UNDELETE only supports physical disk drives.

Cannot operate on the specified drive

You are probably trying to access a network drive. Like most disk utility programs, UNDELETE does not support network drives.

Delete tracking file not found

The UNDELETE command line contained the /DT parameter, and UNDELETE does not have delete tracking information available. You must use the /DOS parameter and attempt to restore the files as recorded by DOS.

Enter the first character of the file name

UNDELETE is attempting to recover the file specified and needs you to type in the original first character of the filename.

The file cannot be recovered

The file specified has been overwritten on disk and cannot be recovered. You must use your backup copy of the file.

Only some of the clusters for this file are available.
Do you want to recover this file with only the available clusters? (Y/N)

UNDELETE is telling you that part of the file specified has been overwritten on disk. UNDELETE can, however, restore part of the file. If you want UNDELETE to restore as much of the file as it can, type **Y**. If you want UNDELETE to ignore the file, type **N**.

UNFORMAT

Restores an erased or formatted disk

UNFORMAT is a DOS 5 or later command that works in conjunction with the MIRROR command (in DOS 5) to rebuild an inadvertently formatted disk or a disk damaged by inadvertently entering the RECOVER command.

Format

```
UNFORMAT D:[/J][/P]
UNFORMAT D:[/L][/TEST][/P]
UNFORMAT D:[/L][/PARTN]
```

D: The disk drive letter of the disk to rebuild.

/J Verifies that the current root directory and file allocation table match the last file created by MIRROR. This parameter does not rebuild the disk. This switch requires DOS 5.

/L Lists each file and subdirectory UNFORMAT finds.

/TEST Displays how UNFORMAT will rebuild the disk, without actually executing the command.

/P Sends screen output to the printer, as well as to the screen.

/PARTN Restores a corrupted partition table using the file MIRROR previously recorded to a floppy disk. This switch requires DOS 5.

Notes

UNFORMAT cannot restore a disk formatted with the /U parameter.

UNFORMAT will always attempt to use the information previously stored by the MIRROR command. If the MIRROR file is not available or is very old, you can direct UNFORMAT to try to recover the disk using only the current file allocation table and root directory. This method, however, is much less reliable.

Using the DOS FORMAT command, create a bootable floppy disk and copy to the disk the UNFORMAT, UNDELETE, RESTORE, and DOSBACK commands. Label the disk as Emergency Disk and place the disk in a safe location.

Examples

The following DOS 5 command uses UNFORMAT to restore a corrupted partition table for drive C:

```
A> UNFORMAT C: /PARTN <ENTER>
```

In this case, UNFORMAT will prompt you to place the floppy disk containing the correct partition table information saved by MIRROR into a floppy drive.

The following UNFORMAT command uses the /TEST parameter to determine whether UNFORMAT can rebuild a formatted floppy disk in drive A:

```
C:\> UNFORMAT A: /TEST <ENTER>
```

Messages

Are you SURE you want to do this?
If so type YES; anything else cancels? No

UNFORMAT gives you the opportunity to cancel the current operation. UNFORMAT normally displays this message before it performs an operation that will actually update the disk. To continue the operation, type **YES** and press ENTER. To cancel the operation, type anything but **YES**.

Cannot process network drive

The drive specified in the UNFORMAT command line is a network disk. Like most disk utilities, UNFORMAT does not support network drives.

CAUTION!! This attempts to recover all files lost after a FORMAT assuming you have not been using MIRROR. This method cannot guarantee complete recovery of your files.

The DOS 5 UNFORMAT command line probably contains the /L parameter that disables UNFORMAT's use of the file MIRROR.FIL created by the MIRROR command. If you don't use the MIRROR file to rebuild the disk, you have a greater chance of losing files.

VER

Displays the DOS version number

The VER command displays the current DOS version number.

Format

```
VER
```

Notes

DOS version numbers are comprised of a major and minor version number. For DOS 3.2, 3 is the major version number, and 2 is the minor version number.

Examples

The following command tells VER to display the current version number:

```
C:\> VER <ENTER>
```

For DOS 3.3, the output is

```
IBM Personal Computer DOS Version 3.30
```

Likewise, for DOS 6, the output is

```
MS-DOS Version 6.00
```

VERIFY

Enables or disables disk verification

Periodically, a disk drive may not correctly record the information on disk as DOS intended. Although rare, such occurrences can leave incorrect data on your disk. If you enable disk I/O verification by issuing the VERIFY command, DOS will double-check the data it writes to disk by rereading each sector and comparing it with the original data. If a discrepancy exists, DOS will detect it. However, because DOS must reread each sector that it writes to disk, disk verification slows down the speed of operation.

Format

```
VERIFY[ON | OFF]
```

ON The parameter that enables DOS disk verification.

OFF The parameter that disables DOS disk verification.

Notes

If you invoke VERIFY without a command-line parameter, it will display its current state, ON or OFF.

The VERIFY command ensures that DOS is successfully recording all the information it is writing to disk. VERIFY does this by comparing each disk sector immediately after it is written to the sector with the information that should have been recorded. Any difference indicates that DOS has recorded data incorrectly.

Although this seems to be a good idea, such disk errors are rare. VERIFY increases the overhead on your system by writing and rereading multiple sectors. Most users should use the default value of

```
C:\> VERIFY OFF <ENTER>
```

Remember, if you are concerned about a file copy, add the /V parameter to the COPY or XCOPY commands:

```
C:\> COPY FILENAME.EXT A: /V <ENTER>
```

DOS will verify the contents of the file as it is recorded to disk.

Examples

The following command enables disk I/O verification:

```
C:\> VERIFY ON <ENTER>
```

Invoking VERIFY without command-line parameters causes VERIFY to display its current state:

```
C:\> VERIFY <ENTER>
VERIFY is on.
C:\>
```

Message

Must specify ON or OFF

You have invoked the VERIFY command specifying a command-line parameter other than ON or OFF. Reissue the command with no command-line parameters (to display the current verify setting) or with either the word ON (to enable disk verification) or OFF (to disable disk verification).

VOL

Displays a disk volume label

The VOL command displays a disk volume label (or name) for the drive specified. DOS volume labels are 11-character names assigned to a disk.

Format

```
VOL[drive:]
```

drive: The disk drive that contains the disk for which VOL is to display the disk volume label. If you do not specify this parameter, VOL uses the default drive.

Notes

Disk labels can consist of the same characters as DOS filenames. To assign a volume label, use the LABEL command.

If you are using DOS 4 or later, the VOL command also displays the disk's optional serial number.

Examples

In the following example, VOL displays the disk volume label of the disk contained in the current drive:

```
C:\> VOL <ENTER>
Volume in drive C is DOSDISK
```

If you are using DOS 4 or later, VOL will also display the disk serial number, as shown here:

```
C:\> VOL <ENTER>
Volume in drive C is DOS
Volume Serial Number is 4E13-1342
```

Message

Volume in drive *n* has no label

VOL is simply telling you that DOS has never assigned a label to the disk in the drive specified. To assign a label to a disk, use the LABEL command.

VSAFE

Installs memory-resident virus detection software

The DOS 6 MSAV scans your computer's disk and memory for known viruses. MSAV only performs a one-time virus search. The VSAFE command, however, installs memory-resident software that detects and intercepts operations frequently performed by destructive viruses. The best virus defense is to use both VSAFE and MSAV.

Format

```
VSAFE  [/OptionNumber[+|-] [...]] [/Ax]
       [/Cx] [/D] [/N] [/NE] [/NX] [/U]
```

/OptionNumber[+|-] Directs VSAFE to turn on or off detection of the following options:

1 Intercepts low-level formatting operations. Default: On
2 Intercepts a program's attempt to become memory resident. Default: Off
3 Intercepts disk write operations. Default: Off
4 Intercepts file operations that open executable files. Default: On
5 Checks all disks for boot sector viruses. Default: On
6 Intercepts attempts to modify the boot record or partition table. Default: On
7 Intercepts attempts to change a floppy disk's boot record. Default: Off
8 Intercepts attempts to change an executable file. Default: On

/Ax Directs VSAFE to use the key specified for ALT key activation. For example, /AX selects ALT-X.

/Cx Directs VSAFE to use the key specified for CTRL key activation. For example /CX selects CTRL-X.

/D Directs VSAFE to turn off checksum operations.

/N Allows network drivers to be loaded after VSAFE is active.

/NE Directs VSAFE not to monitor expanded memory.

/NX Directs VSAFE not to monitor extended memory.

/U Removes VSAFE from memory.

Notes

The VSAFE program consumes approximately 22Kb of memory. If you direct VSAFE to monitor a specific operation, such as a program attempting to become memory resident, and such an event occurs, VSAFE will sound a bell and display a message describing the attempted operation.

To prevent the operation, select Stop and the offending program will end. To allow the operation select Continue. If you select Boot, your system will restart.

You can use VSAFE within Windows; however, you must first place the following entry in your WIN.INI file:

```
LOAD=C:\DOS\WNTSRMAN.EXE
```

VSAFE requires DOS 6

Example

The following command loads VSAFE into memory:

```
C:\> VSAFE <ENTER>
```

If you press VSAFE's activation keyboard combination (ALT-V is the default), VSAFE will display its menu. To enable or disable VSAFE's monitoring of specific events, type the event's number. VSAFE will display a check mark after each event it will monitor. Press ESC to hide the menu.

XCOPY

Copies source files and subdirectories to a target disk

The XCOPY command copies one or more files providing extended capabilities not available with the COPY command.

Format

```
XCOPY source_file_spec[target_file_spec]
    [/A][/D:mm-dd-yy][/E][/M][/P][/S][/V][/W]
```

source_file_spec The complete DOS file specification for the source files to be copied by XCOPY.

target_file_spec The destination name for the files copied by XCOPY.

/A Copies only files that have the archive bit set.

/D:mm-dd-yy Copies only those files created on or after the specified date.

/E Places subdirectories on the target disk if the subdirectory is currently empty.

/M Functions like the /A parameter; however, /M tells XCOPY to clear each file's archive bit as it copies the file.

/P Tells XCOPY to prompt the following before copying each file.

```
FILENAME.EXT (Y/N)?
```

/S Copies the contents of lower-level subdirectories to the target location.

/V Compares the contents of the target file and the source file to verify that the file copy was successful.

/W Tells XCOPY to prompt the following before beginning.

```
Press any key to begin copying file(s)
```

Notes

The XCOPY command copies source files and subdirectories to a target destination. Unlike COPY and DISKCOPY, the XCOPY command copies files contained in DOS subdirectories on a selective basis. Many users invoke XCOPY to repair disk fragmentation or use it as a system backup mechanism.

Because the XCOPY command provides many more capabilities than the standard COPY command, many users are using XCOPY in place of COPY for their file manipulation. In the future, the use of XCOPY should continue to grow, and its functionality should increase.

Examples

The following XCOPY command copies the files in the current directory of drive C to the disk in drive A. Using the /S parameter, the command creates an identical copy of the directory on the floppy drive, including all of the files that reside in subdirectories below the current directory:

```
C:\> XCOPY *.*  A:\ /S <ENTER>
```

The following command copies all of the files in the current directory that have been created or modified since December 31, 1993, to the disk in drive A:

```
C:\> XCOPY *.* /D:12/31/93 A: <ENTER>
```

To use XCOPY to copy the entire contents of a fixed disk to floppy disks, first set the attribute of each file on the fixed disk to indicate that each requires a backup, as shown here:

```
C:\> ATTRIB +A \*.* /S <ENTER>
```

Next, issue this command:

```
C:\> XCOPY \*.* A:\ /M /E /S <ENTER>
```

XCOPY will begin transferring files to the floppy disk, maintaining the existing disk structure. When the target disk becomes full, simply insert a new floppy disk in drive A and again invoke the command:

```
C:\> XCOPY \*.* A:\ /M /E /S <ENTER>
```

XCOPY will continue where it left off, because it has been clearing the archive bit on each file it successfully copies to the target disk.

Messages

Access denied

Either the source file specified in the XCOPY command is locked by another application, or the target filename has been marked as read-only.

Cannot COPY from a reserved device

The XCOPY command line specifies a device such as COM1 as the source of the file copy. XCOPY does not support this type of operation. Use the COPY command.

Cannot COPY to a reserved device

The XCOPY command line specifies a device such as COM1 as the target of the file copy. XCOPY does not support this type of operation. Use the COPY command.

Cannot perform cyclic copy

The XCOPY command line contains the /S parameter, and the source and target specifications are such that a subdirectory specified in the source is also a subdirectory in the target.

Does *FILENAME* specify a file name or directory name on the target (F=file, D=directory)?

XCOPY could not determine whether the target of the file copy operation should be a filename or directory. If you want XCOPY to write all of the information to a file, simply type **F** and press ENTER. If you want XCOPY to create a subdirectory, type **D** and press ENTER.

drive:path.ext (Y/N)?

You have invoked the XCOPY command with the /P parameter, which performs selective file copying. If you don't want to copy this file, simply respond with **N**. To copy the file, type **Y** and press ENTER.

Appendix A

Exit Status Values

This appendix lists the exit status values that DOS commands provide for use with IF ERRORLEVEL in your batch files.

BACKUP

Status Value	Meaning
0	Successful backup operation
1	No files found to back up
2	File-sharing conflicts prevented a complete backup
3	Backup operation ended by user CTRL-C
4	Backup operation ended due to fatal error

DEFRAG

Status Value	Meaning
0	Successful defragmentation
1	Internal error in processing
2	No free clusters—DEFRAG needs at least 1

Status Value	Meaning
3	User termination via CTRL-C
4	General error in processing
5	Disk read error
6	Disk write error
7	Cluster allocation error;use CHKDSK /F
8	Memory allocation error
9	Insufficient memory

DISKCOMP

Status Value	Meaning
0	Disks are the same
1	Disks are not the same
2	Comparison ended by user CTRL-C
3	Comparison ended due to fatal disk error
4	Initialization error prevented comparison

DISKCOPY

Status Value	Meaning
0	Successful disk copy operation
1	Non-fatal disk read/write error
2	Disk copy operation ended by user CTRL-C
3	Disk copy ended due to fatal processing error
4	Initialization error prevented copy operation

FORMAT

Status Value	Meaning
0	Successful format operation
3	Format process ended by user CTRL-C

Status Value	Meaning
4	Format process ended due to fatal processing error
5	Format processes ended by user at the prompt "Proceed with Format (Y/N)?"

GRAFTABL

Status Value	Meaning
0	Character set successfully loaded
1	Previously loaded character set replaced
2	File access error prevented character set loading
3	Invalid command-line parameter prevented character set loading
4	Incorrect DOS version

KEYB

Status Value	Meaning
0	Keyboard template successfully loaded
1	Invalid keyboard code, code page, or command syntax prevented template loading
2	Keyboard definition file was not found
4	Error accessing CON device prevented template loading
5	Requested code page was not previously prepared

REPLACE

Status Value	Meaning
0	Successful file replacement
2	Source files not found
3	Source or destination directory path not found
5	Access to destination file denied

Status Value	Meaning
8	Insufficient RAM to execute REPLACE
11	Invalid command-line syntax
15	Invalid disk drive specified
22	Incorrect DOS version (not provided by MS-DOS)

RESTORE

Status Value	Meaning
0	Successful file restoration
1	No files found to restore
2	File sharing conflict prevented complete restoration (not provided by MS-DOS)
3	Restore operation ended by user CTRL-C
4	Restore operation ended by fatal error

SETVER

Status Value	Meaning
0	Successful version assignment
1	Invalid command switch
2	Invalid filename specified
3	Insufficient RAM for SETVER to execute
4	Invalid version number specified
5	Entry specified not found in version table
6	System files not found
7	Invalid drive letter specified
8	Too many command-line parameters specified
9	One or more command-line parameters missing
10	Error reading system files
11	System file version table is corrupt

Status Value	Meaning
12	System files specified don't support a version table
13	Insufficient version table space for new entry
14	Error writing to system files

XCOPY

Status Value	Meaning
0	Successful file copy operation
1	No files found to copy
2	File copy operation terminated by user CTRL-C
4	Initialization error prevented file copy operation
5	Disk write error during copy operation

Appendix B

ASCII and Extended ASCII Character Sets

Dec	Hex	ASCII Symbol	Control Code	Ctrl Key	Dec	Hex	ASCII Symbol	Control Code	Ctrl Key
0	00		NUL	^@	16	10	►	DLE	^P
1	01	☺	SOH	^A	17	11	◄	DC1	^Q
2	02	☻	STX	^B	18	12	↕	DC2	^R
3	03	♥	ETX	^C	19	13	!!	DC3	^S
4	04	♦	EOT	^D	20	14	¶	DC4	^T
5	05	♣	ENQ	^E	21	15	§	NAK	^U
6	06	♠	ACK	^F	22	16	▬	SYN	^V
7	07	•	BEL	^G	23	17	↨	ETB	^W
8	08	◘	BS	^H	24	18	↑	CAN	^X
9	09	○	HT	^I	25	19	↓	EM	^Y
10	0A	◙	LF	^J	26	1A	→	SUB	^Z
11	0B	♂	VT	^K	27	1B	←	ESC	^[
12	0C	♀	FF	^L	28	1C	∟	FS	^\
13	0D	♪	CR	^M	29	1D	↔	GS	^]
14	0E	♫	SO	^N	30	1E	▲	RS	^
15	0F	☼	SI	^O	31	1F	▼	US	^_

Dec	Hex	ASCII Symbol	Dec	Hex	ASCII Symbol
32	20		71	47	G
33	21	!	72	48	H
34	22	"	73	49	I
35	23	#	74	4A	J
36	24	$	75	4B	K
37	25	%	76	4C	L
38	26	&	77	4D	M
39	27	'	78	4E	N
40	28	(	79	4F	O
41	29	)	80	50	P
42	2A	*	81	51	Q
43	2B	+	82	52	R
44	2C	,	83	53	S
45	2D	-	84	54	T
46	2E	.	85	55	U
47	2F	/	86	56	V
48	30	0	87	57	W
49	31	1	88	58	X
50	32	2	89	59	Y
51	33	3	90	5A	Z
52	34	4	91	5B	[
53	35	5	92	5C	\
54	36	6	93	5D	]
55	37	7	94	5E	^
56	38	8	95	5F	_
57	39	9	96	60	`
58	3A	:	97	61	a
59	3B	;	98	62	b
60	3C	<	99	63	c
61	3D	=	100	64	d
62	3E	>	101	65	e
63	3F	?	102	66	f
64	40	@	103	67	g
65	41	A	104	68	h
66	42	B	105	69	i
67	43	C	106	6A	j
68	44	D	107	6B	k
69	45	E	108	6C	l
70	46	F	109	6D	m

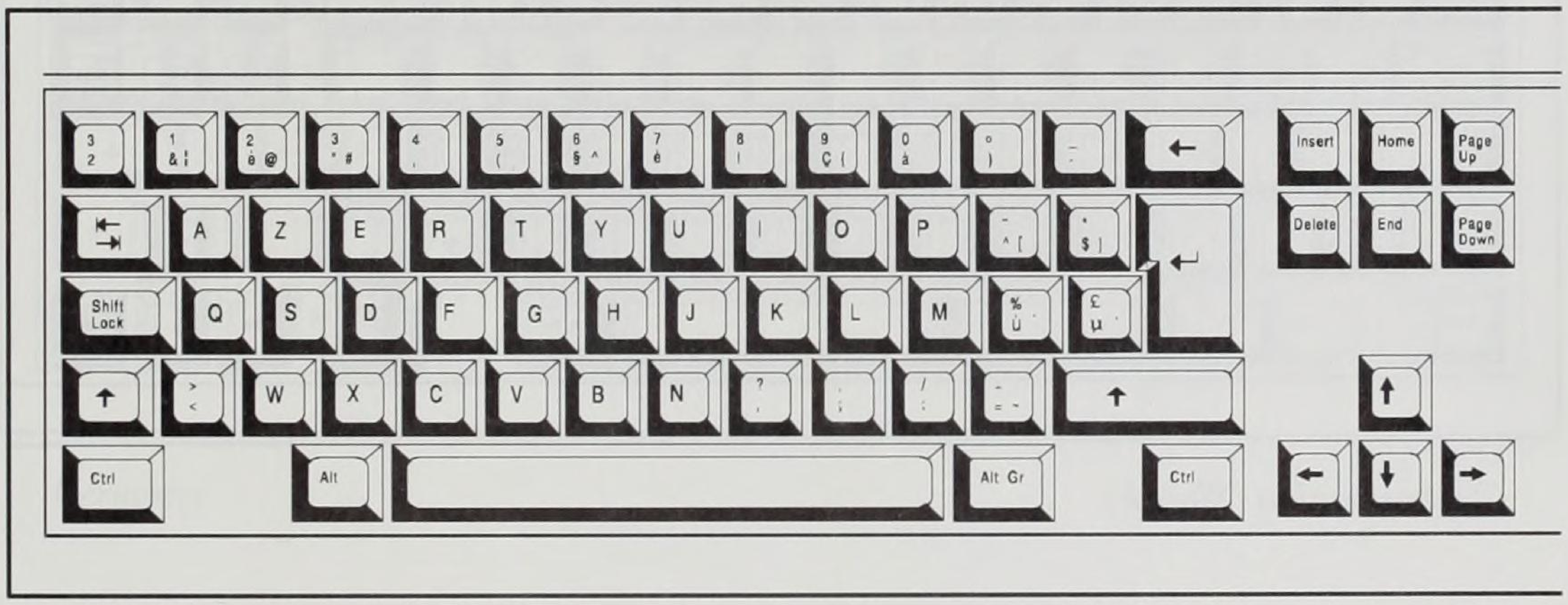
Dec	Hex	ASCII Symbol	Dec	Hex	ASCII Symbol
110	6E	n	149	95	ò
111	6F	o	150	96	û
112	70	p	151	97	ù
113	71	q	152	98	ÿ
114	72	r	153	99	Ö
115	73	s	154	9A	Ü
116	74	t	155	9B	¢
117	75	u	156	9C	£
118	76	v	157	9D	¥
119	77	w	158	9E	Pt
120	78	x	159	9F	ƒ
121	79	y	160	A0	á
122	7A	z	161	A1	í
123	7B	{	162	A2	ó
124	7C	|	163	A3	ú
125	7D	}	164	A4	ñ
126	7E	~	165	A5	Ñ
127	7F	⌂	166	A6	ª
128	80	Ç	167	A7	º
129	81	ü	168	A8	¿
130	82	é	169	A9	⌐
131	83	â	170	AA	¬
132	84	ä	171	AB	½
133	85	à	172	AC	¼
134	86	å	173	AD	¡
135	87	ç	174	AE	<<
136	88	ê	175	AF	>>
137	89	ë	176	B0	░
138	8A	è	177	B1	▒
139	8B	ï	178	B2	▓
140	8C	î	179	B3	│
141	8D	ì	180	B4	┤
142	8E	Ä	181	B5	╡
143	8F	Å	182	B6	╢
144	90	É	183	B7	╖
145	91	æ	184	B8	╕
146	92	Æ	185	B9	╣
147	93	ô	186	BA	║
148	94	ö	187	BB	╗

Dec	Hex	ASCII Symbol	Dec	Hex	ASCII Symbol
188	BC	╝	222	DE	▐
189	BD	╜	223	DF	▀
190	BE	╛	224	E0	α
191	BF	┐	225	E1	β
192	C0	└	226	E2	Γ
193	C1	┴	227	E3	π
194	C2	┬	228	E4	Σ
195	C3	├	229	E5	σ
196	C4	─	230	E6	µ
197	C5	┼	231	E7	τ
198	C6	╞	232	E8	φ
199	C7	╟	233	E9	θ
200	C8	╚	234	EA	Ω
201	C9	╔	235	EB	δ
202	CA	╩	236	EC	∞
203	CB	╦	237	ED	∅
204	CC	╠	238	EE	∈
205	CD	═	239	EF	∩
206	CE	╬	240	F0	≡
207	CF	╧	241	F1	±
208	D0	╨	242	F2	≥
209	D1	╤	243	F3	≤
210	D2	╥	244	F4	⌠
211	D3	╙	245	F5	⌡
212	D4	╘	246	F6	÷
213	D5	╒	247	F7	≈
214	D6	╓	248	F8	°
215	D7	╫	249	F9	•
216	D8	╪	250	FA	·
217	D9	┘	251	FB	√
218	DA	┌	252	FC	η
219	DB	█	253	FD	²
220	DC	▄	254	FE	■
221	DD	▌	255	FF	

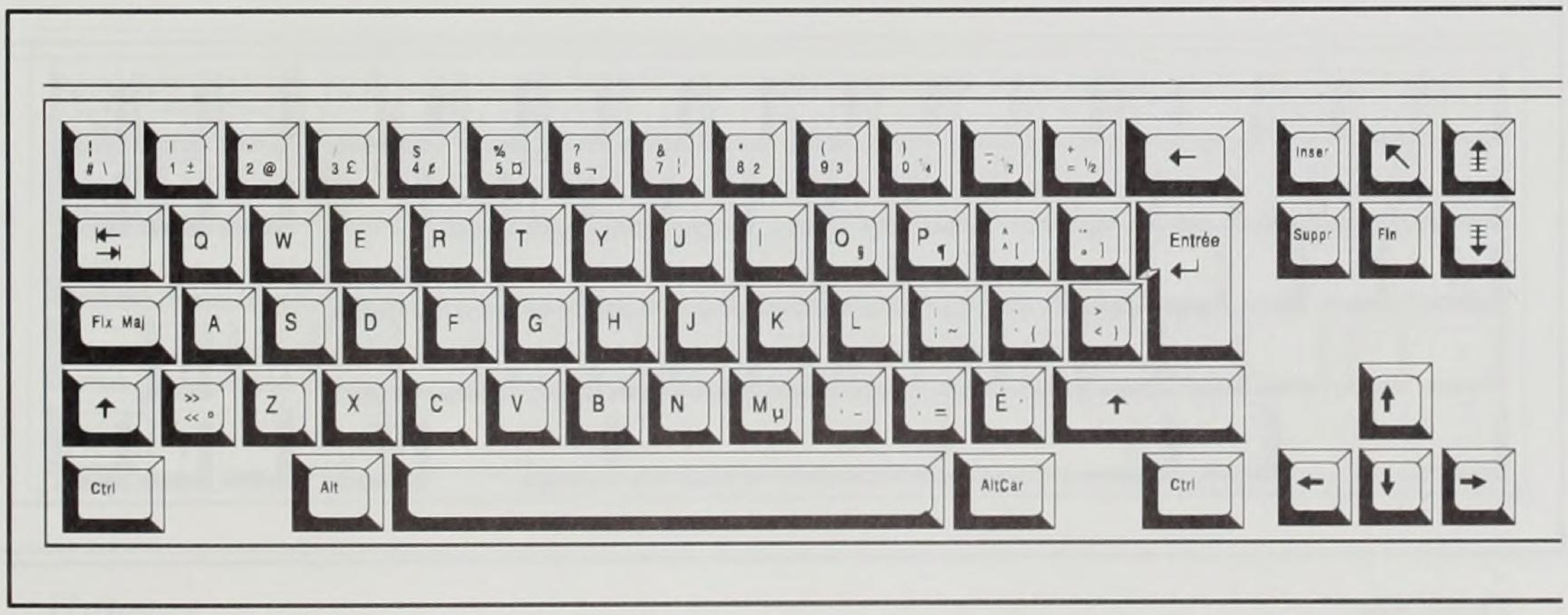
Appendix C

International Keyboard Layouts and Code Page Tables

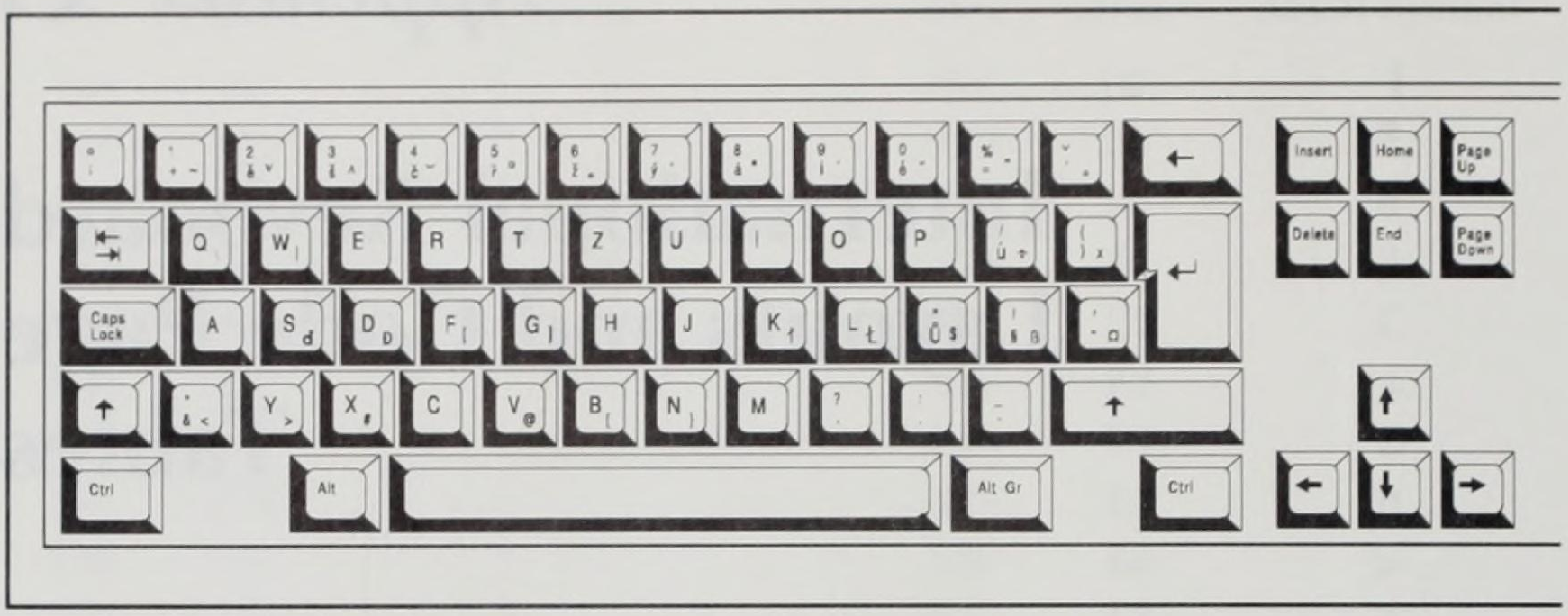
Keyboard Layouts


3 2
1 & ¦
2 é @
3 " #
4 '
5 (
6 § ^
7 è
8 !
9 Ç {
0 à
°)
_ -
A
Z
E
R
T
Y
U
I
O
P
¨ ^ [
* $]
Shift Lock
Q
S
D
F
G
H
J
K
L
M
% ù
£ µ
> <
W
X
C
V
B
N
? ,
. ;
/ :
+ = ~
Ctrl
Alt
Alt Gr
Ctrl
Insert
Home
Page Up
Delete
End
Page Down

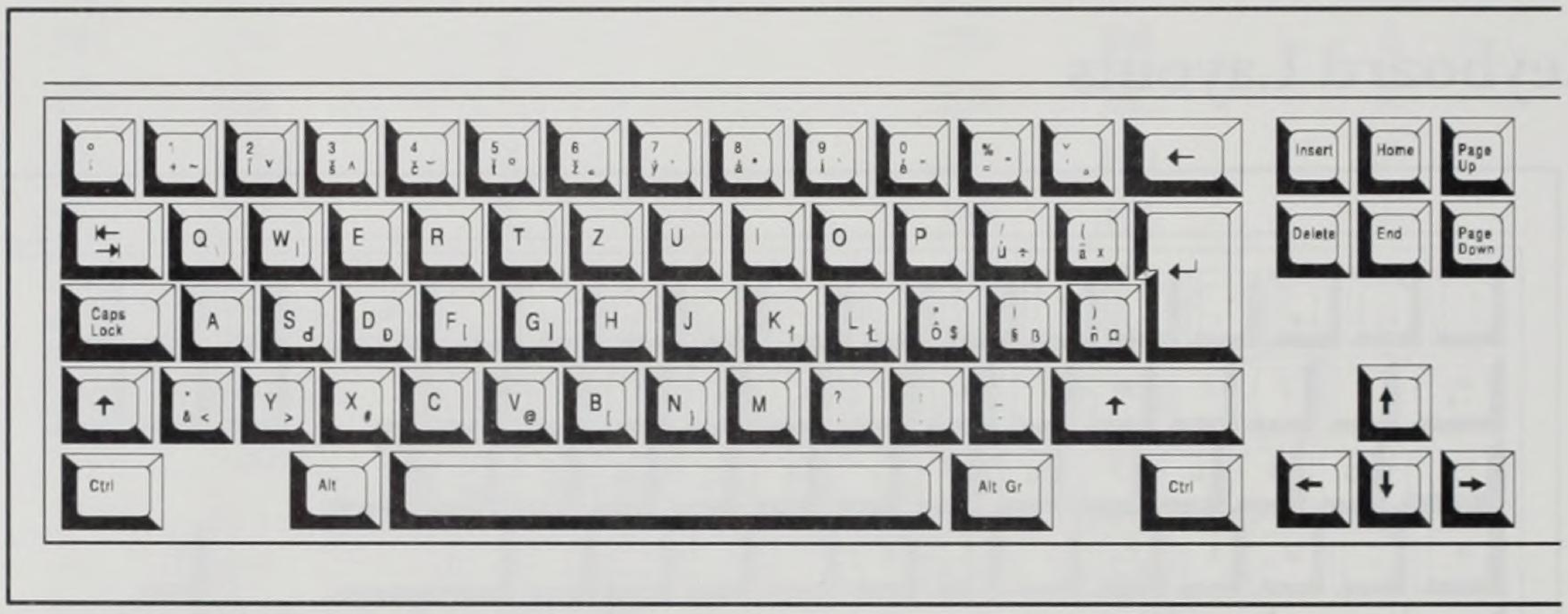
Belgium


| # \
| 1 ±
" 2 @
/ 3 £
$ 4 ¢
% 5 ¤
? 6 ¬
& 7 ¦
* 8 ²
(9 ³
) 0 ¼
_ - ½
+ = ¾
Q
W
E
R
T
Y
U
I
O §
P ¶
^ ^ [
¨ ¸]
Entrée
Fix Maj
A
S
D
F
G
H
J
K
L
: ; ~
" ` {
> < }
>> << °
Z
X
C
V
B
N
M µ
' , -
" . =
É ´
Ctrl
Alt
AltCar
Ctrl
Inser
Suppr
Fin

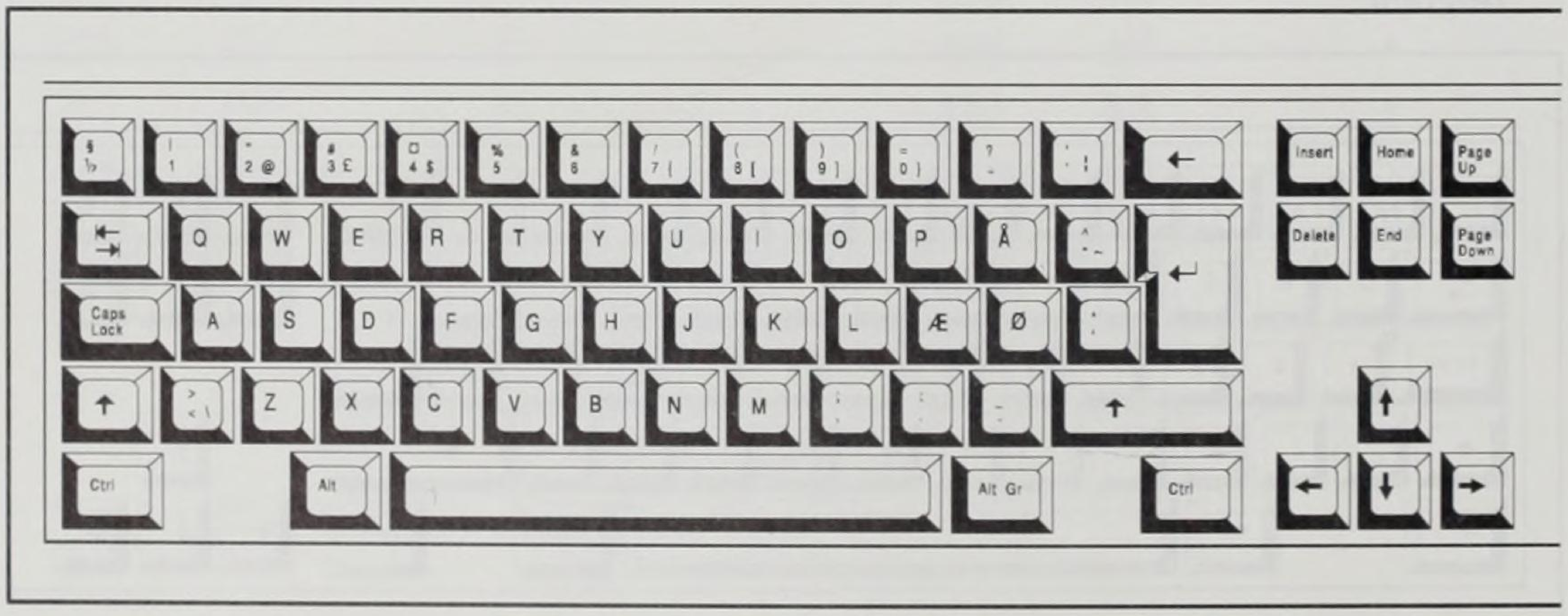
Canada


Insert
Home
Page Up
Delete
End
Page Down
Q W E R T Z U I O P
Caps Lock
A S D F G H J K L
Y X C V B N M
Ctrl
Alt
Alt Gr
Ctrl

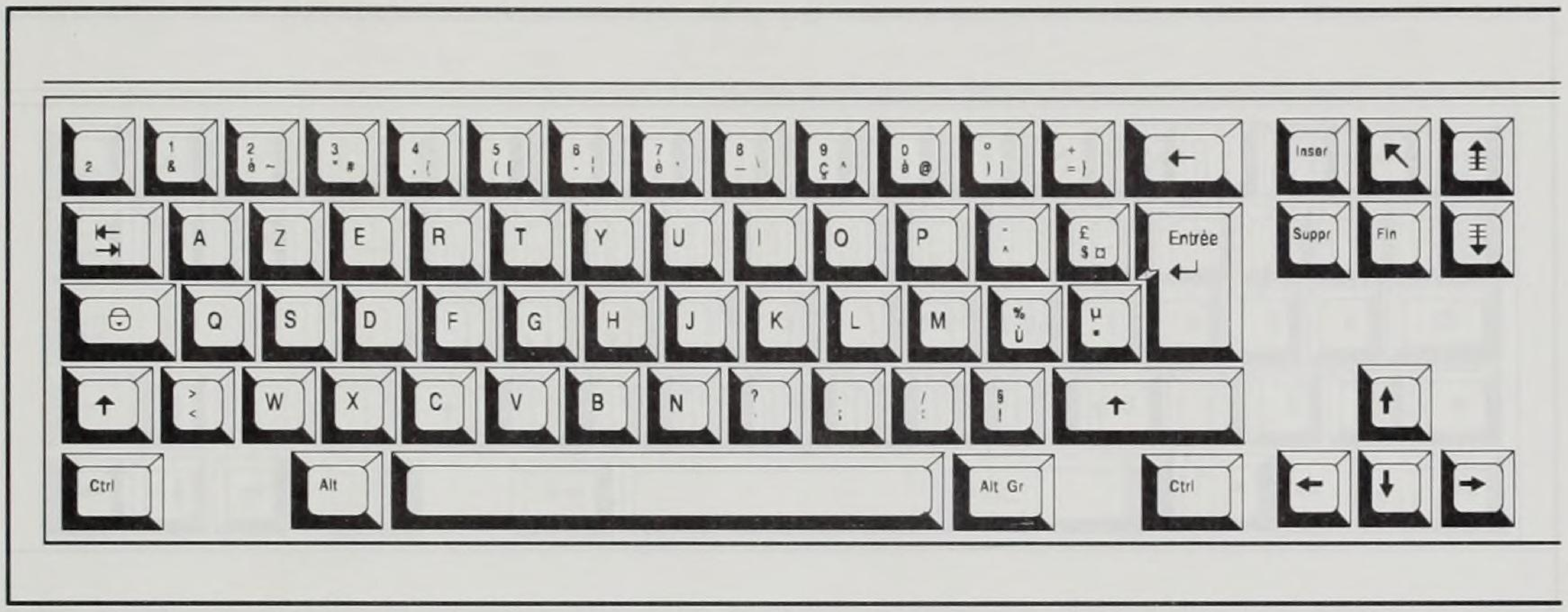
Czechoslovakia (Czech)


Insert
Home
Page Up
Delete
End
Page Down
Q W E R T Z U I O P
Caps Lock
A S D F G H J K L
Y X C V B N M
Ctrl
Alt
Alt Gr
Ctrl

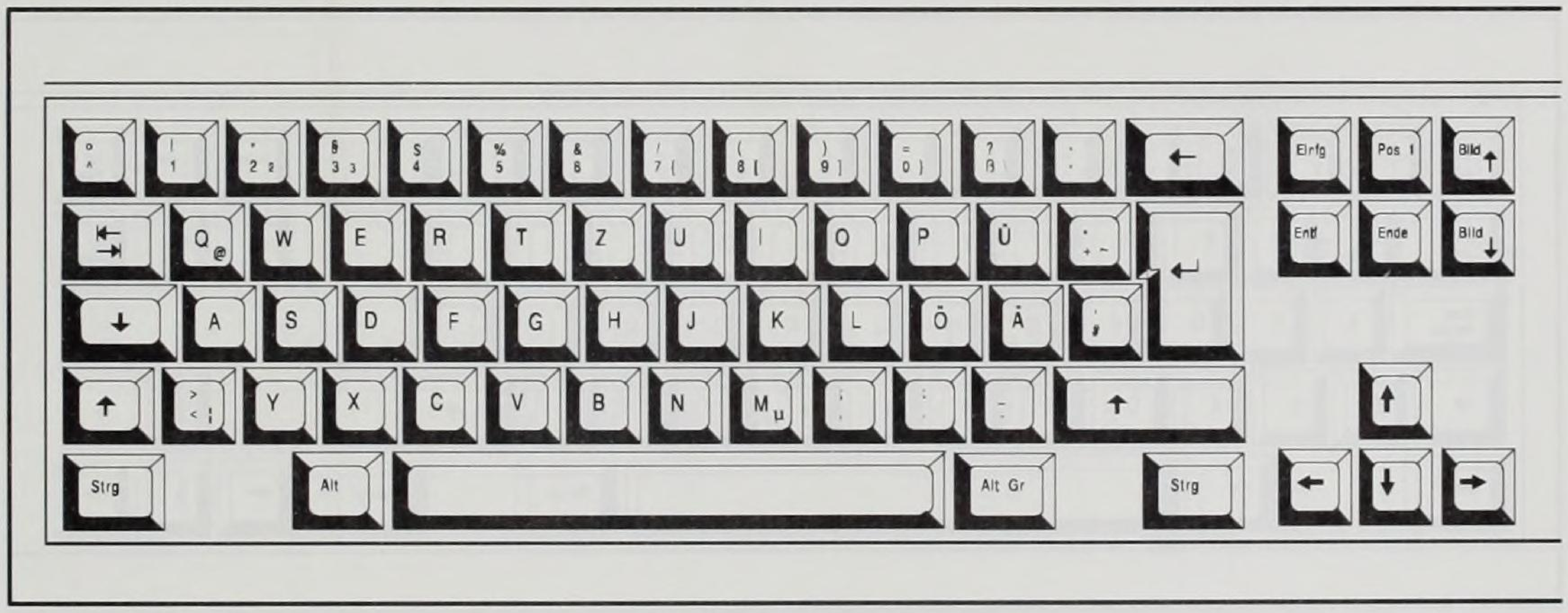
Czechoslovakia (Slovak)


Insert
Home
Page Up
Delete
End
Page Down
Q W E R T Y U I O P Å
Caps Lock
A S D F G H J K L Æ Ø
Z X C V B N M
Ctrl
Alt
Alt Gr
Ctrl

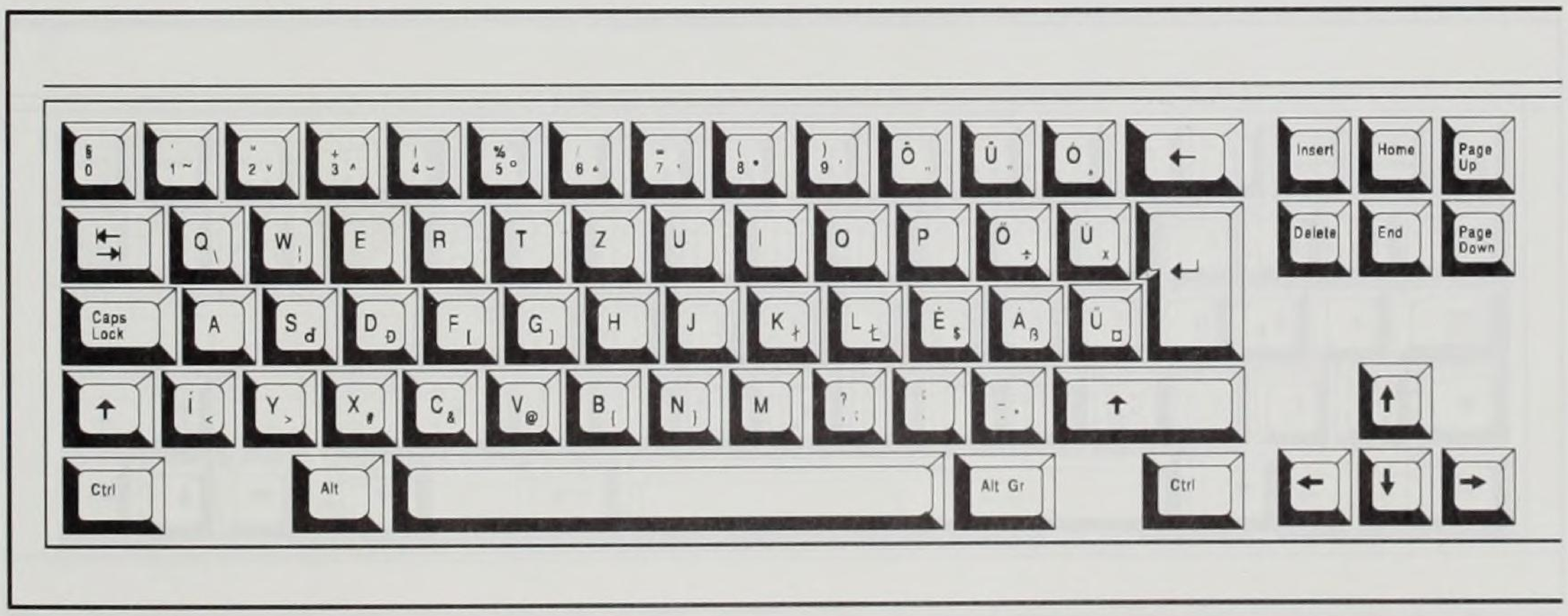
Denmark


A
Z
E
R
T
Y
U
I
O
P
Entrée
Q
S
D
F
G
H
J
K
L
M
W
X
C
V
B
N
Ctrl
Alt
Alt Gr
Ctrl
Inser
Suppr
Fin

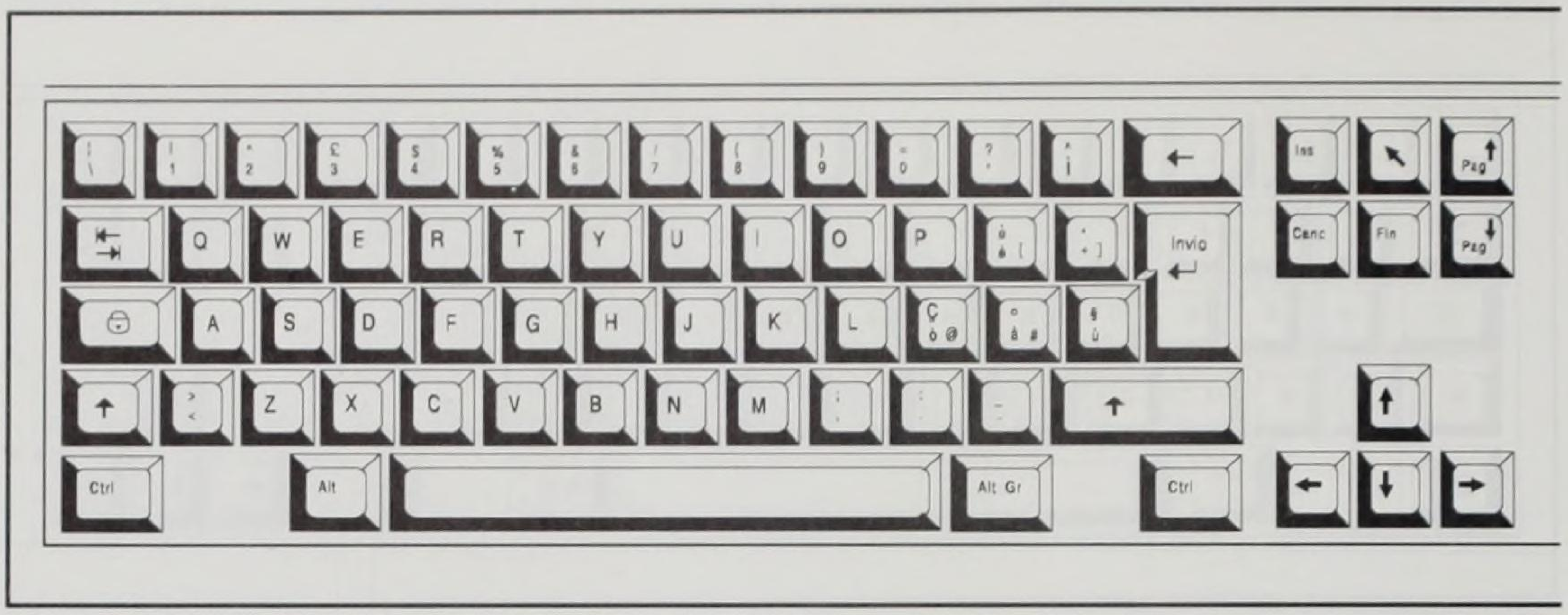
France


Einfg
Pos 1
Bild
Q
W
E
R
T
Z
U
I
O
P
Ü
Entf
Ende
Bild
A
S
D
F
G
H
J
K
L
Ö
Ä
Y
X
C
V
B
N
M
Strg
Alt
Alt Gr
Strg

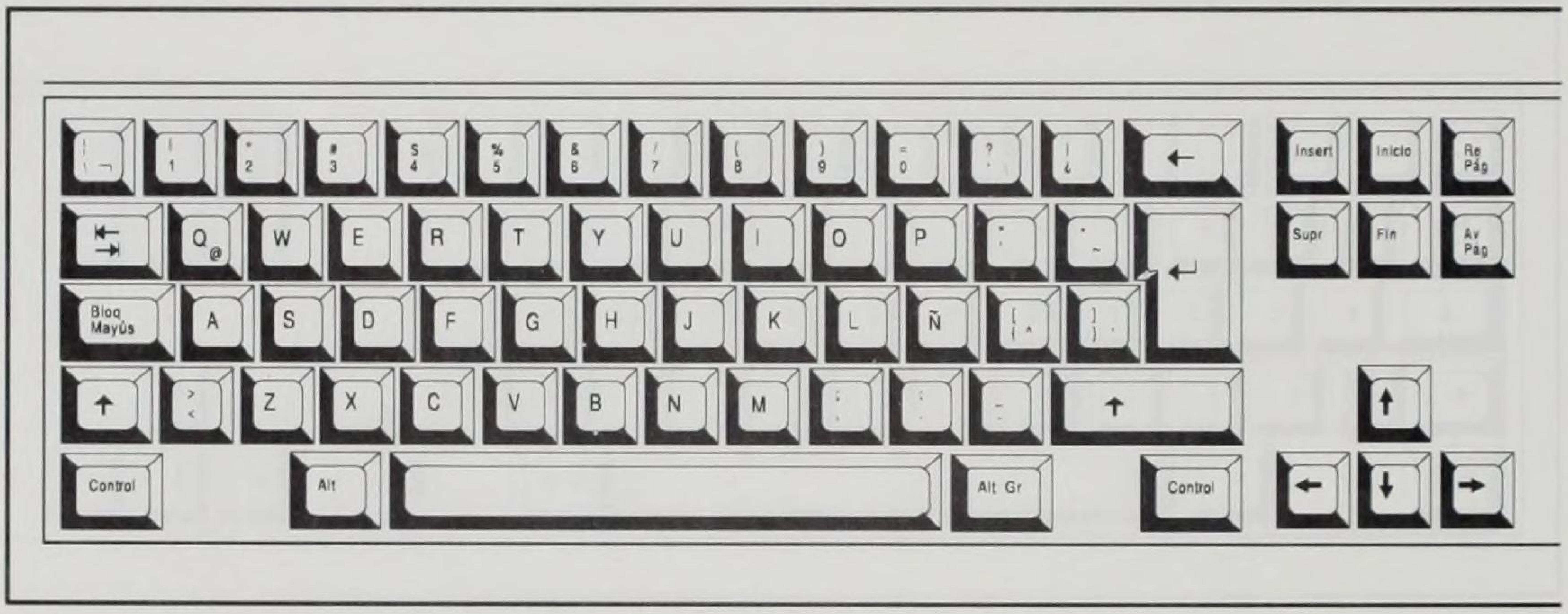
Germany


Ö
Ü
Ó
Insert
Home
Page Up
Q
W
E
R
T
Z
U
I
O
P
Ő
Ú
Delete
End
Page Down
Caps Lock
A
S
D
F
G
H
J
K
L
É
Á
Ű
Í
Y
X
C
V
B
N
M
Ctrl
Alt
Alt Gr
Ctrl

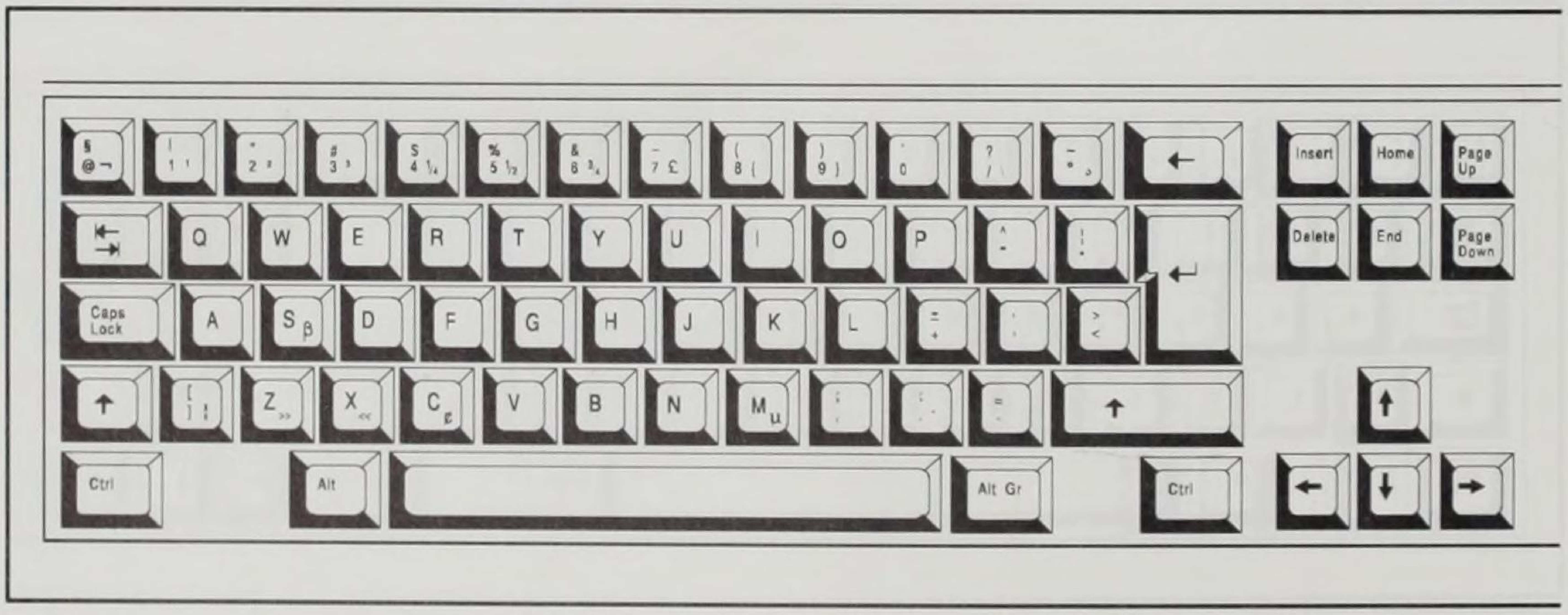
Hungary



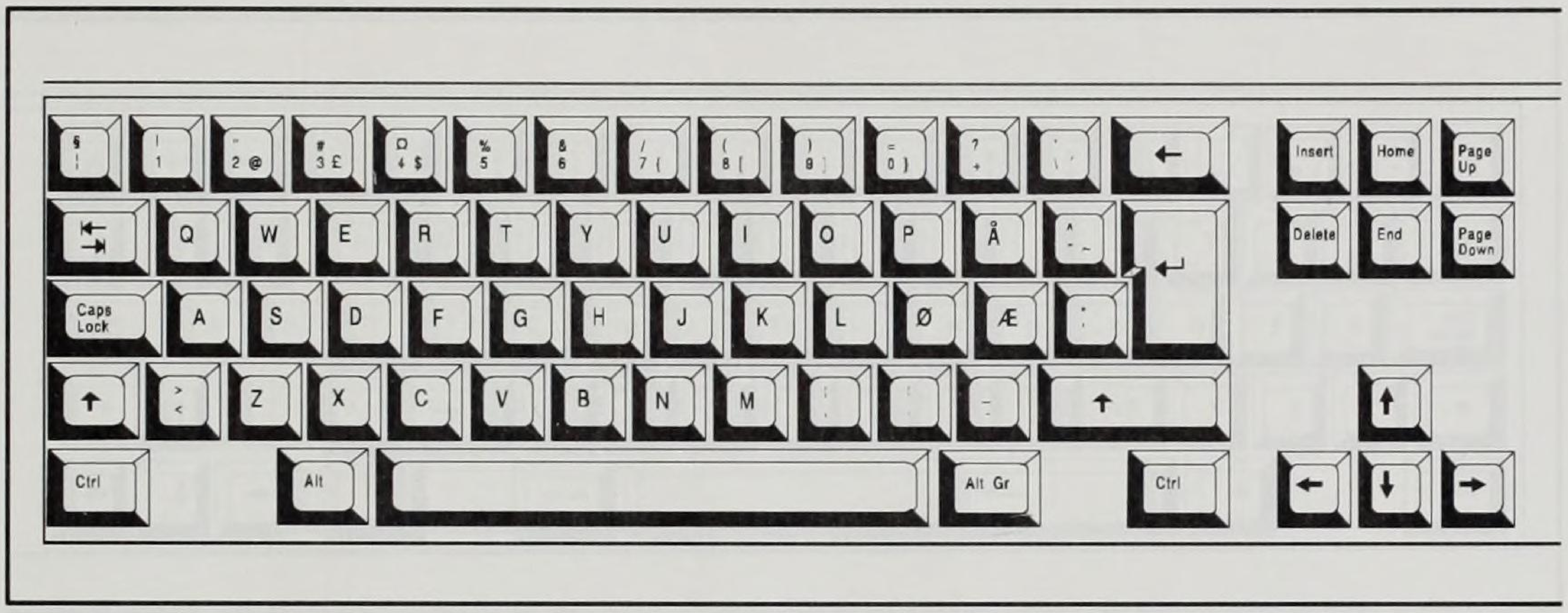
Italy



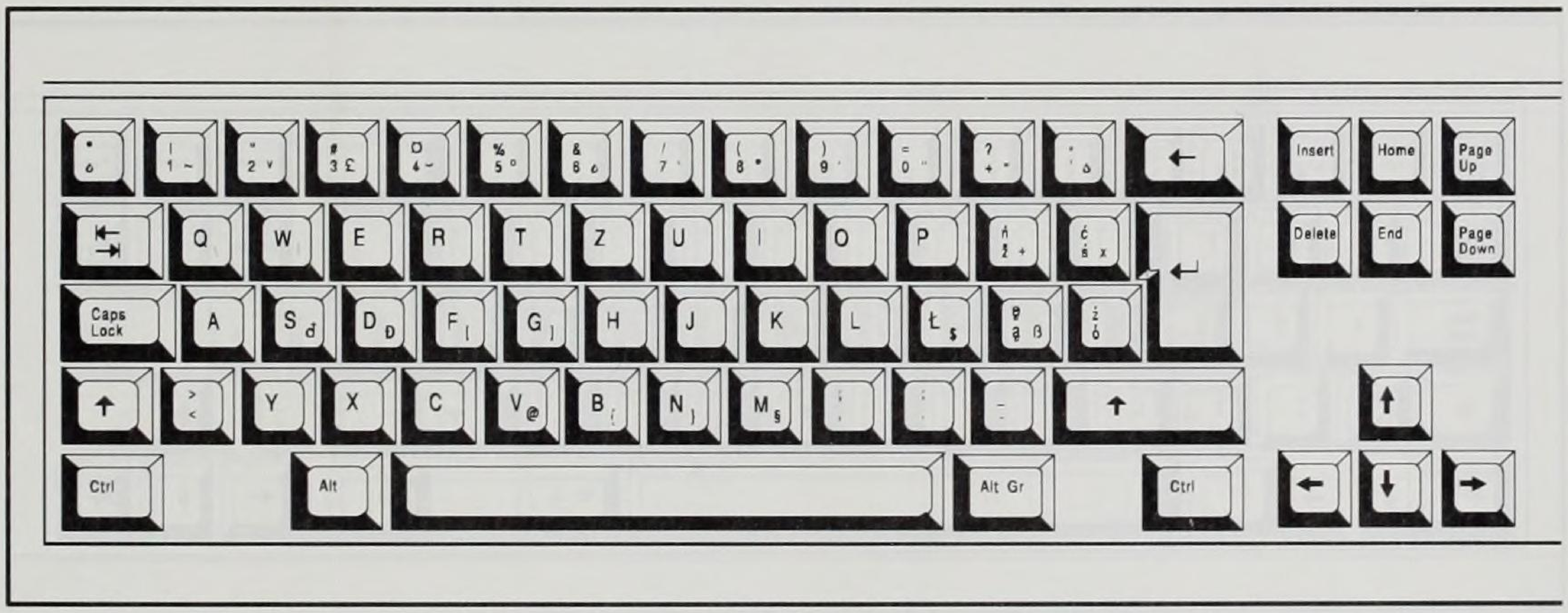
Latin America



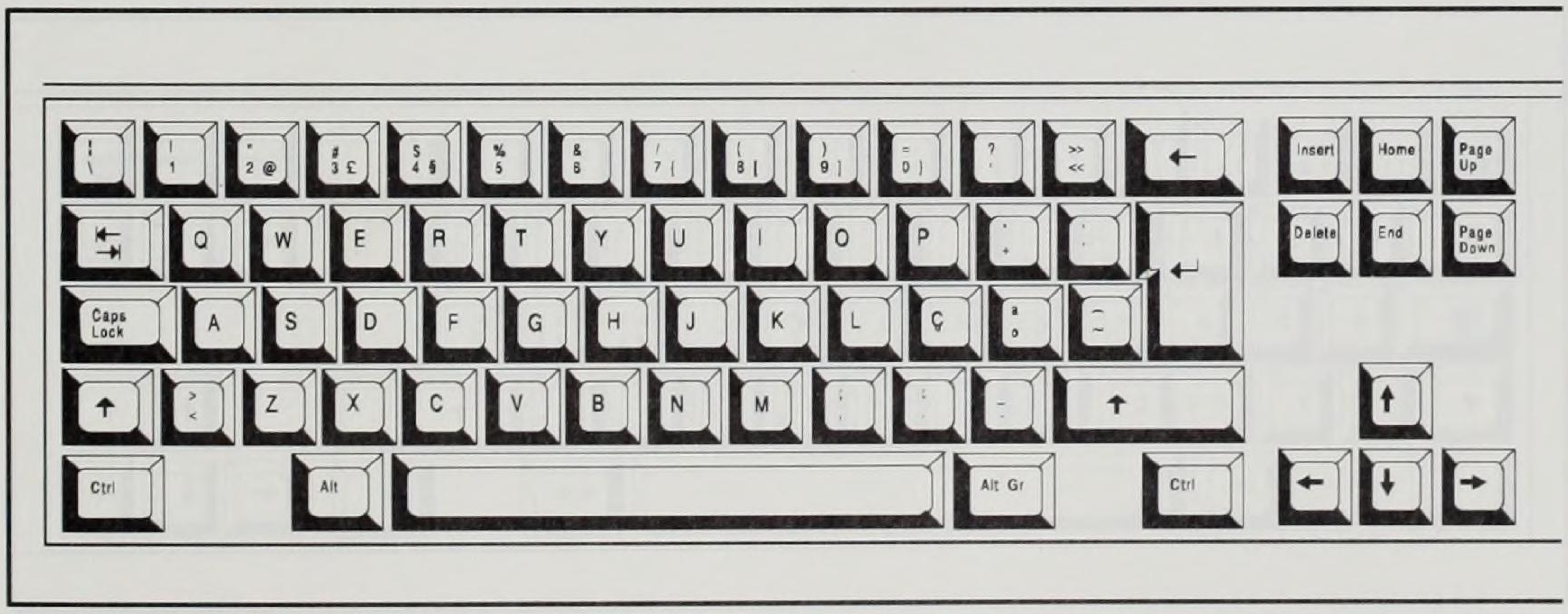
Netherlands


Q W E R T Y U I O P Å
Caps Lock A S D F G H J K L Ø Æ
Z X C V B N M
Ctrl Alt Alt Gr Ctrl
Insert Home Page Up
Delete End Page Down

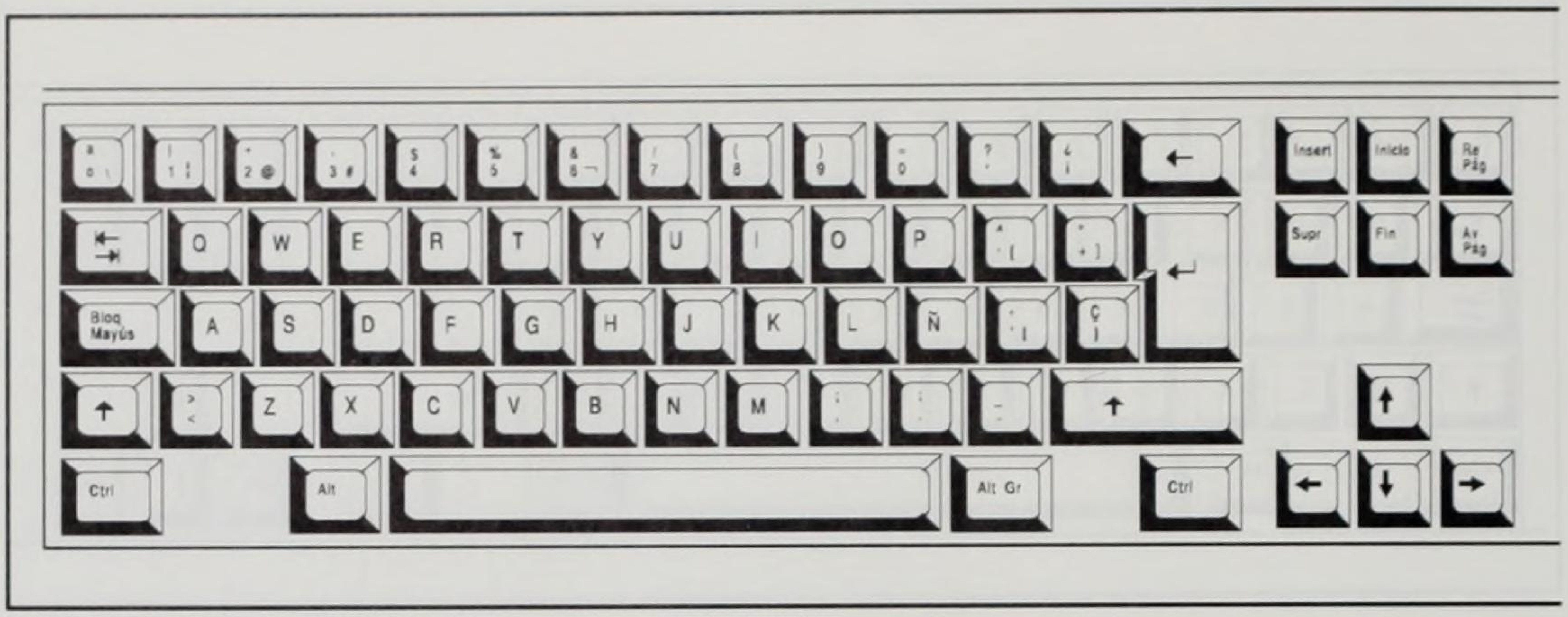
Norway


Q W E R T Z U I O P
Caps Lock A S D F G H J K L Ł
Y X C V B N M
Ctrl Alt Alt Gr Ctrl
Insert Home Page Up
Delete End Page Down

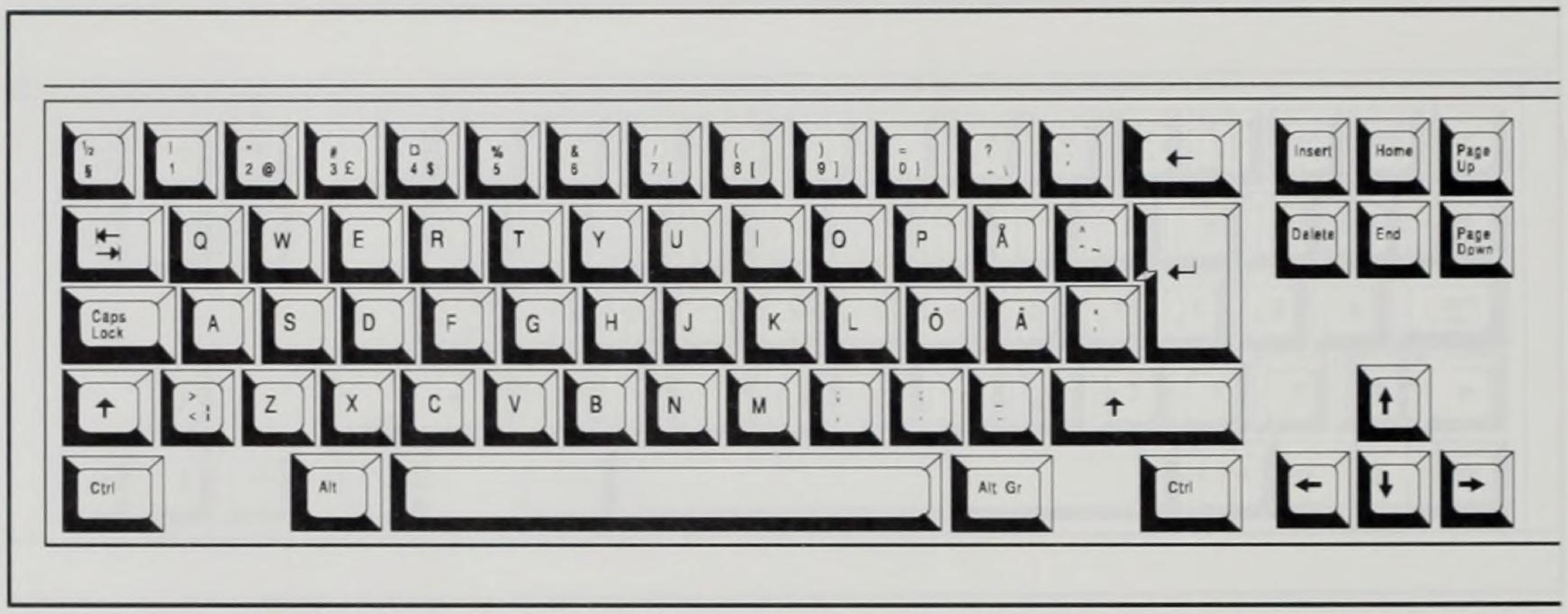
Poland


Q W E R T Y U I O P
Caps Lock A S D F G H J K L Ç
Z X C V B N M
Ctrl Alt Alt Gr Ctrl
Insert Home Page Up
Delete End Page Down

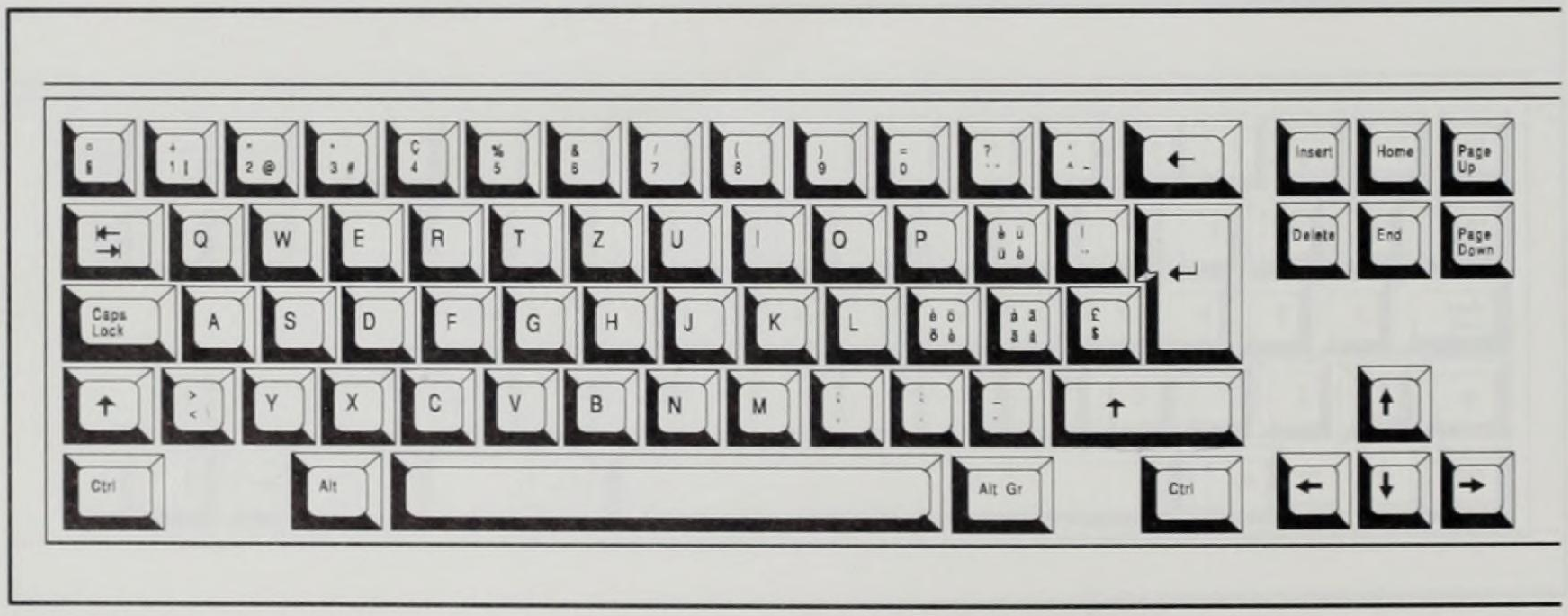
Portugal



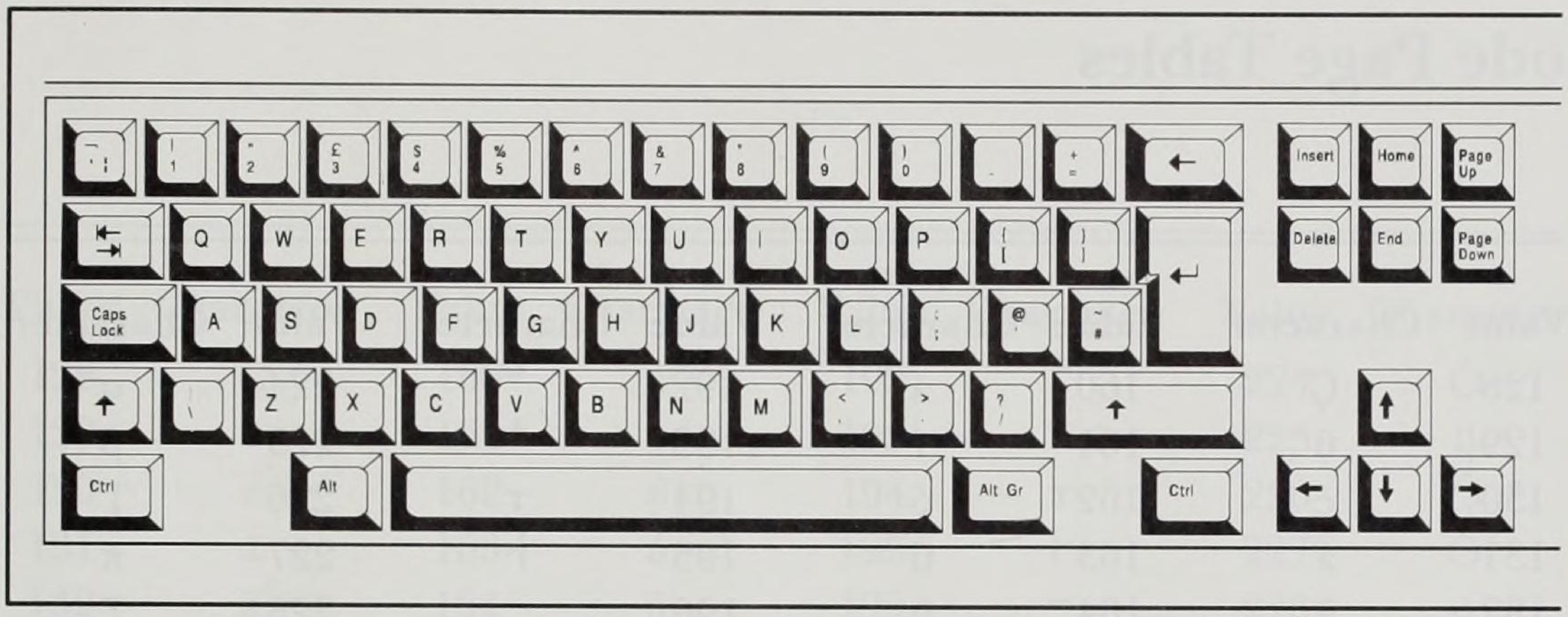
Spain



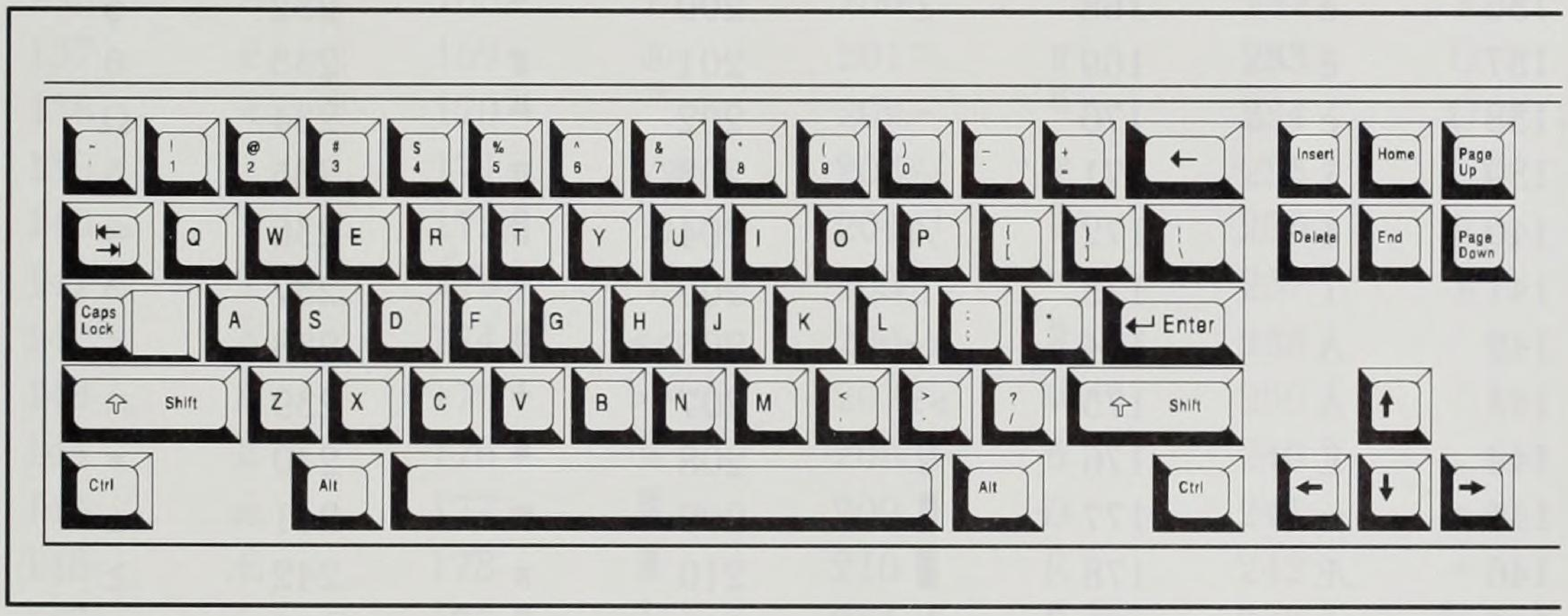
Sweden/Finland



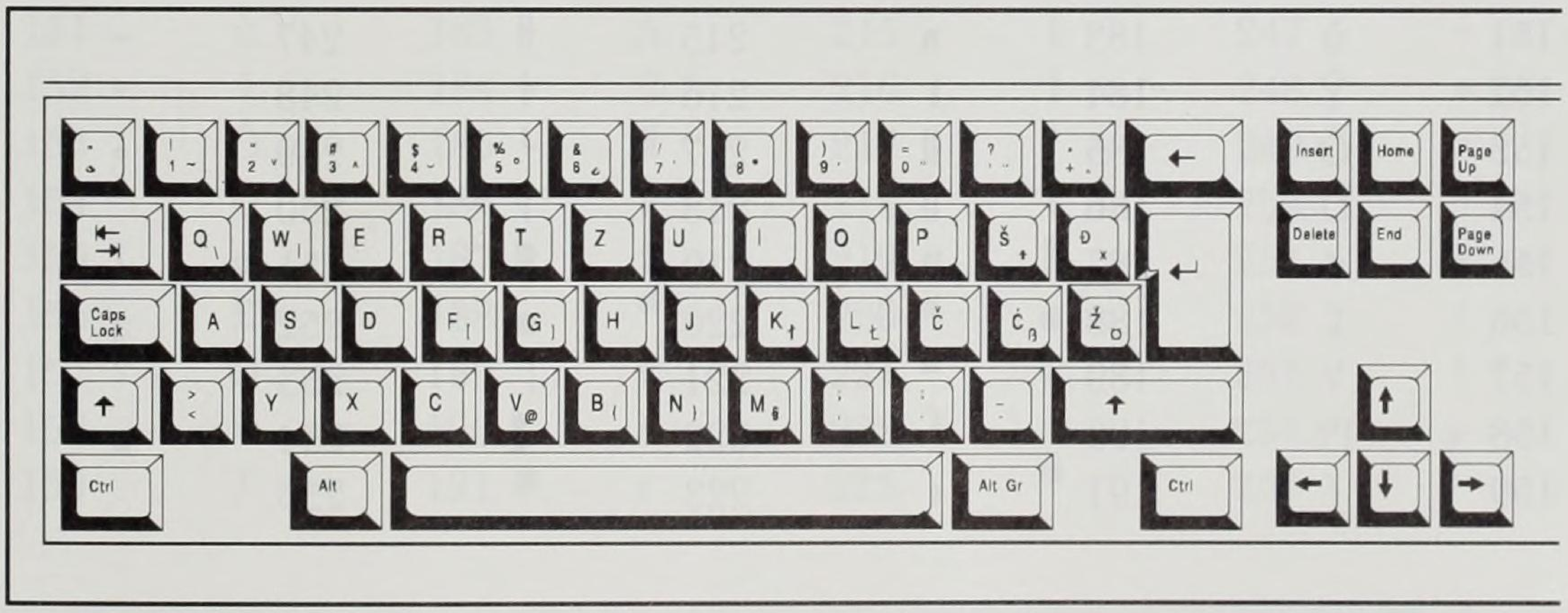
Switzerland


¬ ` ! 1 " 2 £ 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = ←
Insert Home Page Up
Q W E R T Y U I O P { [}] ↵
Delete End Page Down
Caps Lock A S D F G H J K L : ; @ ' ~ #
↑ | \ Z X C V B N M < , > . ? / ↑
↑
Ctrl Alt Alt Gr Ctrl
← ↓ →

United Kingdom


~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = ←
Insert Home Page Up
Q W E R T Y U I O P { [}] | \
Delete End Page Down
Caps Lock A S D F G H J K L : ; " ' ↵ Enter
⇧ Shift Z X C V B N M < , > . ? / ⇧ Shift
↑
Ctrl Alt Alt Ctrl
← ↓ →

United States


! 1 ~ " 2 ˇ # 3 ^ $ 4 ˘ % 5 ° & 6 ˛ / 7 ` (8 ˙) 9 ´ = 0 ˝ ? ' ¨ * + ¸ ←
Insert Home Page Up
Q \ W | E R T Z U I O P Š ÷ Đ × ↵
Delete End Page Down
Caps Lock A S D F [G] H J K ł L Ł Č Ć ß Ž ¤
↑ > < Y X C V @ B { N } M § ; , : . _ - ↑
↑
Ctrl Alt Alt Gr Ctrl
← ↓ →

Yugoslavia

Code Page Tables

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	á	192	└	224	α
129	ü	161	í	193	┴	225	β
130	é	162	ó	194	┬	226	Γ
131	â	163	ú	195	├	227	π
132	ä	164	ñ	196	─	228	Σ
133	à	165	Ñ	197	┼	229	σ
134	å	166	ª	198	╞	230	μ
135	ç	167	º	199	╟	231	τ
136	ê	168	¿	200	╚	232	ϕ
137	ë	169	⌐	201	╔	233	θ
138	è	170	¬	202	╩	234	Ω
139	ï	171	½	203	╦	235	δ
140	î	172	¼	204	╠	236	∞
141	ì	173	¡	205	═	237	∅
142	Ä	174	<<	206	╬	238	∈
143	Å	175	>>	207	╧	239	∩
144	É	176	░	208	╨	240	≡
145	æ	177	▒	209	╤	241	±
146	Æ	178	▓	210	╥	242	≥
147	ô	179	│	211	╙	243	≤
148	ö	180	┤	212	╘	244	⌠
149	ò	181	╡	213	╒	245	⌡
150	û	182	╢	214	╓	246	÷
151	ù	183	╖	215	╫	247	≈
152	Ÿ	184	╕	216	╪	248	°
153	Ö	185	╣	217	┘	249	•
154	Ü	186	║	218	┌	250	·
155	¢	187	╗	219	█	251	√
156	£	188	╝	220	▄	252	η
157	¥	189	╜	221	¦	253	²
158	Pt	190	╛	222	▐	254	■
159	ƒ	191	┐	223	▀	255	

Table C-1. *Code Page 437, United States*

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	<<	206	╬	238	¯
143	Å	175	>>	207	¤	239	´
144	É	176	░	208	ð	240	.
145	æ	177	▒	209	Đ	241	±
146	Æ	178	▓	210	Ê	242	=
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	•
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	

Table C-2. Code Page 850, Multilingual

Code Page Tables (*continued*)

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ń
132	ä	164	Ą	196	─	228	ń
133	ů	165	ą	197	┼	229	ň
134	ć	166	Ž	198	Ă	230	Š
135	ç	167	ž	199	ă	231	š
136	ł	168	Ę	200	╚	232	Ŕ
137	ë	169	ę	201	╔	233	Ú
138	Ő	170	¬	202	╩	234	ŕ
139	ő	171	ź	203	╦	235	Ü
140	î	172	Č	204	╠	236	ý
141	Ź	173	Ş	205	═	237	Ý
142	Ä	174	»	206	╬	238	ţ
143	Ć	175	«	207	¤	239	´
144	É	176	░	208	đ	240	–
145	Ĺ	177	▒	209	Đ	241	˝
146	i	178	▓	210	Ď	242	˛
147	ô	179	│	211	Ë	243	˜
148	ö	180	┤	212	ď	244	˘
149	Ľ	181	Á	213	Ň	245	§
150	ľ	182	Â	214	Í	246	÷
151	Ś	183	Ě	215	Î	247	ɔ
152	ś	184	Ş	216	ě	248	•
153	Ö	185	╣	217	┘	249	··
154	Ü	186	║	218	┌	250	·
155	Ť	187	╗	219	█	251	ű
156	ť	188	╝	220	▄	252	Ř
157	Ł	189	Ż	221	Ţ	253	ř
158	×	190	ż	222	U̇	254	■
159	č	191	┐	223	▀	255	

Table C-3. *Code Page 852, Slavic (Latin II)*

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	á	192	└	224	α
129	ü	161	í	193	┴	225	β
130	é	162	ó	194	┬	226	Γ
131	â	163	ú	195	├	227	π
132	ã	164	ñ	196	─	228	Σ
133	à	165	Ñ	197	┼	229	σ
134	Á	166	ª	198	╞	230	µ
135	ç	167	º	199	╟	231	τ
136	ê	168	¿	200	╚	232	Φ
137	Ê	169	Ò	201	╔	233	θ
138	è	170	¬	202	╩	234	Ω
139	ì	171	½	203	╦	235	δ
140	Ô	172	¼	204	╠	236	∞
141	ì	173	¡	205	═	237	∅
142	Ã	174	<<	206	╬	238	∈
143	Â	175	>>	207	╧	239	∩
144	É	176	░	208	╨	240	≡
145	À	177	▒	209	╤	241	±
146	È	178	▓	210	╥	242	≥
147	ô	179	│	211	╙	243	≤
148	õ	180	┤	212	╘	244	⌠
149	ò	181	╡	213	╒	245	⌡
150	Ú	182	╢	214	╓	246	÷
151	ù	183	╖	215	╫	247	≈
152	Ì	184	╕	216	╪	248	°
153	Õ	185	╣	217	┘	249	•
154	Ü	186	║	218	┌	250	.
155	¢	187	╗	219	█	251	√
156	£	188	╝	220	▄	252	η
157	Ù	189	╜	221	¦	253	^{2}
158	Pt	190	╛	222	▐	254	■
159	Ó	191	┐	223	▀	255	

Table C-4. *Code Page 860, Portuguese*

Code Page Tables (*continued*)

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	¦	192	└	224	α
129	ü	161	´	193	┴	225	β
130	é	162	ó	194	┬	226	Γ
131	â	163	ú	195	├	227	π
132	Â	164	¨	196	─	228	Σ
133	à	165	¸	197	┼	229	σ
134	¶	166	³	198	╞	230	µ
135	ç	167	¯	199	╟	231	τ
136	ê	168	Î	200	╚	232	Φ
137	ë	169	⌐	201	╔	233	θ
138	è	170	¬	202	╩	234	Ω
139	ï	171	½	203	╦	235	δ
140	î	172	¼	204	╠	236	∞
141	=	173	¾	205	═	237	ø
142	À	174	<<	206	╬	238	∈
143	§	175	>>	207	╧	239	∩
144	É	176	░	208	╨	240	≡
145	È	177	▒	209	╤	241	±
146	Ê	178	▓	210	╥	242	≥
147	ô	179	│	211	╙	243	≤
148	Ë	180	┤	212	╘	244	⌠
149	Ï	181	╡	213	╒	245	⌡
150	û	182	╢	214	╓	246	÷
151	ú	183	╖	215	╫	247	≈
152	¤	184	╕	216	╪	248	°
153	Ô	185	╣	217	┘	249	•
154	Ü	186	║	218	┌	250	·
155	¢	187	╗	219	█	251	√
156	£	188	╝	220	▄	252	η
157	Ù	189	╜	221	¦	253	²
158	Û	190	╛	222	▐	254	■
159	ƒ	191	┐	223	▀	255	

Table C-5. *Code Page 863, French Canadian*

Value	Character	Value	Character	Value	Character	Value	Character
128	Ç	160	á	192	└	224	α
129	ü	161	í	193	┴	225	β
130	é	162	ó	194	┬	226	Γ
131	â	163	ú	195	├	227	π
132	ä	164	ñ	196	─	228	Σ
133	à	165	Ñ	197	┼	229	σ
134	å	166	ª	198	╞	230	µ
135	ç	167	º	199	╟	231	τ
136	ê	168	¿	200	╚	232	Φ
137	ë	169	⌐	201	╔	233	θ
138	è	170	¬	202	╩	234	Ω
139	ï	171	½	203	╦	235	δ
140	î	172	¼	204	╠	236	∞
141	ì	173	¡	205	═	237	ø
142	Ä	174	<<	206	╬	238	∈
143	Å	175	¤	207	╧	239	∩
144	É	176	░	208	╨	240	≡
145	æ	177	▒	209	╤	241	±
146	Æ	178	▓	210	╥	242	≥
147	ô	179	│	211	╙	243	≤
148	ö	180	┤	212	╘	244	⌠
149	ò	181	╡	213	╒	245	⌡
150	û	182	╢	214	╓	246	÷
151	ù	183	╖	215	╫	247	≈
152	ÿ	184	╕	216	╪	248	°
153	Ö	185	╣	217	┘	249	•
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	√
156	£	188	╝	220	▄	252	η
157	Ø	189	╜	221	▌	253	²
158	Pt	190	╛	222	▐	254	■
159	ƒ	191	┐	223	▀	255	

Table C-6. Code Page 865, Norwegian

Appendix D

Upgrading to DOS 6

When you purchase DOS 6 with a new computer, your computer manufacturer should install DOS 6 on to your hard disk for you. If you are currently using a different version of DOS, you can purchase and install the DOS 6 upgrade as discussed in this appendix.

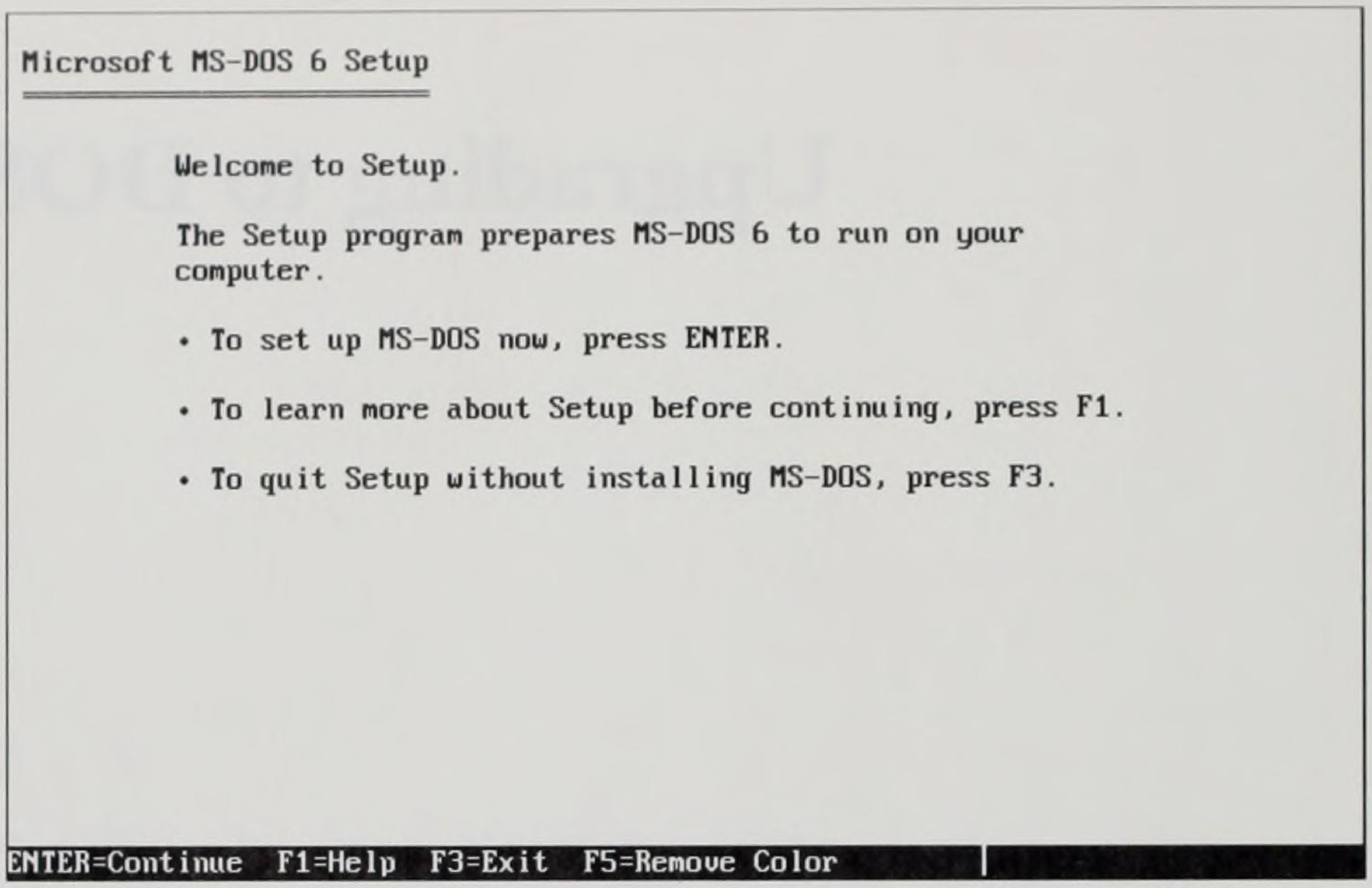
Performing the DOS 6 Upgrade

The DOS 6 upgrade ships on several 3 1/2- or 5 1/4-inch disks. During the upgrade, the Setup program will ask you to insert one or two floppy disks that you have labelled Uninstall 1 and Uninstall 2. Therefore, you need two unused floppy disks available before you begin the upgrade.

To begin the installation, place Disk 1 of the upgrade disks into drive A, select drive A as the current drive, and issue the SETUP command as shown here:

```
A:\> SETUP <ENTER>
```

Setup will display the screen shown in Figure D-1, which welcomes you to the DOS 6 Setup.

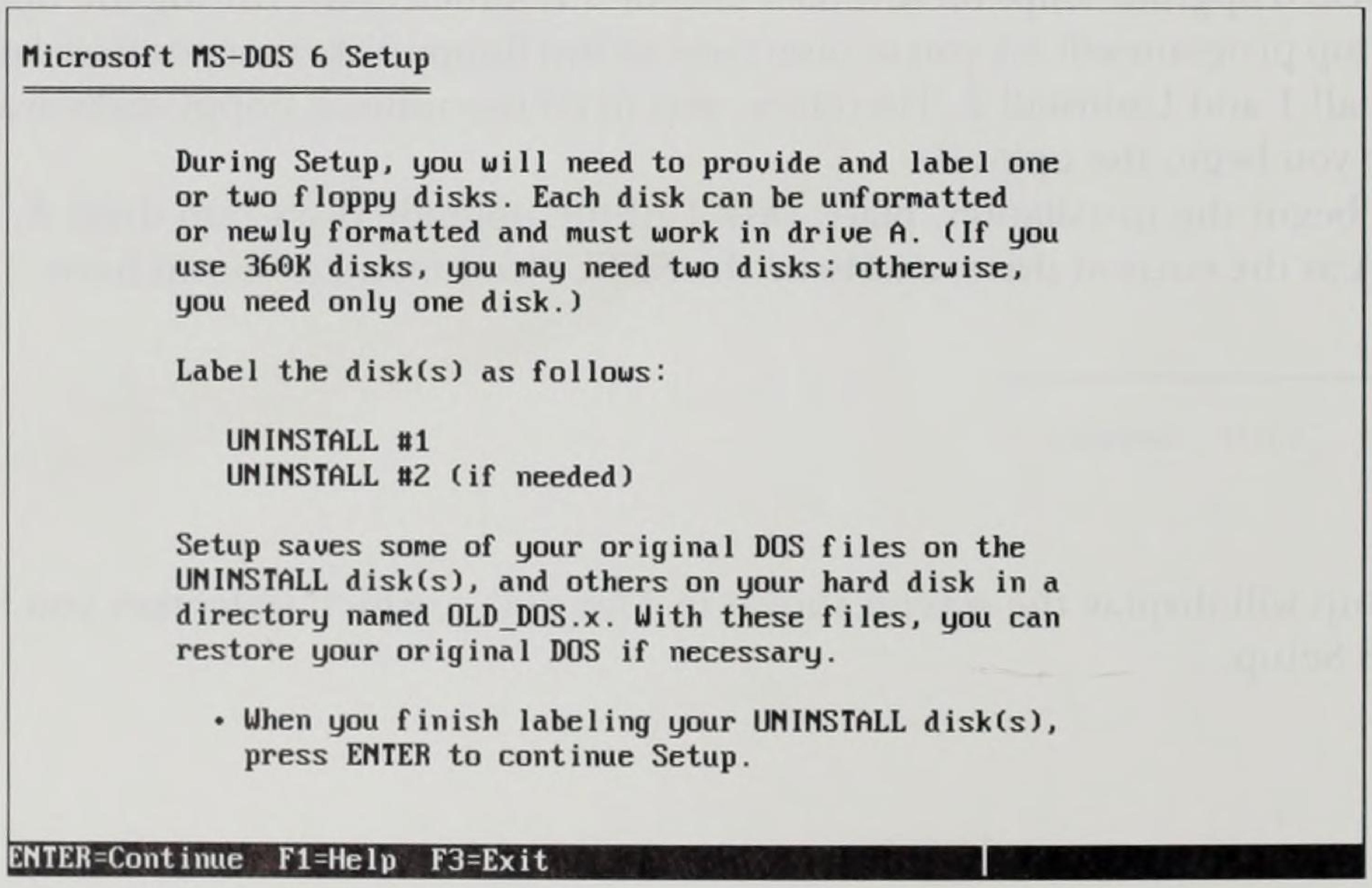


```
Microsoft MS-DOS 6 Setup

          Welcome to Setup.

          The Setup program prepares MS-DOS 6 to run on your
          computer.

          • To set up MS-DOS now, press ENTER.

          • To learn more about Setup before continuing, press F1.

          • To quit Setup without installing MS-DOS, press F3.
```

Figure D-1. *The Welcome to DOS 6 Setup screen*

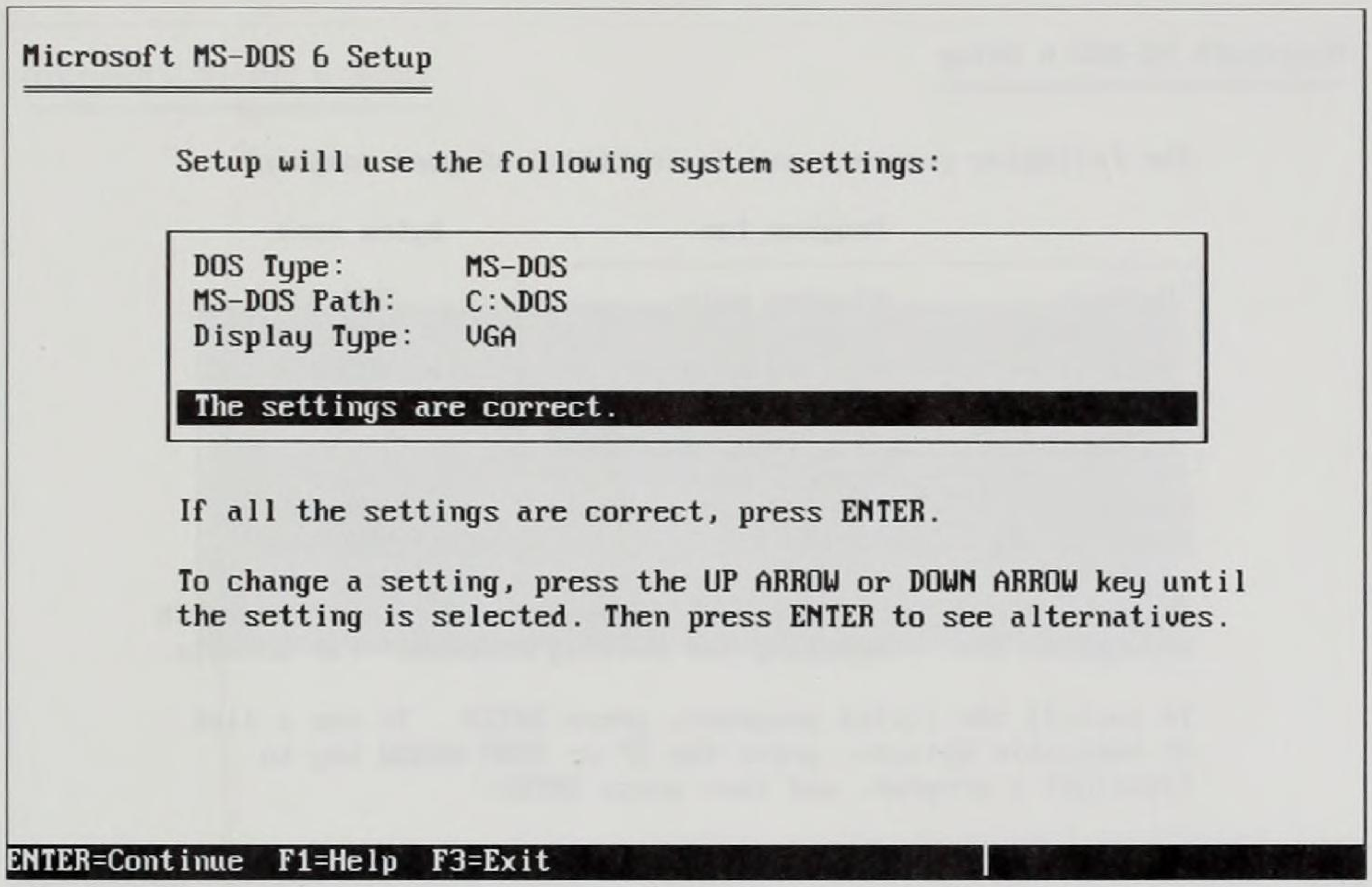
To continue the upgrade, press ENTER. Setup will display the screen shown in Figure D-2, informing you that it might require two Uninstall disks.

Press ENTER to continue the upgrade. Setup will then display a screen containing specifics about your system, as shown in Figure D-3.



```
Microsoft MS-DOS 6 Setup

          During Setup, you will need to provide and label one
          or two floppy disks. Each disk can be unformatted
          or newly formatted and must work in drive A. (If you
          use 360K disks, you may need two disks; otherwise,
          you need only one disk.)

          Label the disk(s) as follows:

             UNINSTALL #1
             UNINSTALL #2 (if needed)

          Setup saves some of your original DOS files on the
          UNINSTALL disk(s), and others on your hard disk in a
          directory named OLD_DOS.x. With these files, you can
          restore your original DOS if necessary.

             • When you finish labeling your UNINSTALL disk(s),
               press ENTER to continue Setup.
```

Figure D-2. *The Setup Uninstall Overview screen*



```
Microsoft MS-DOS 6 Setup

        Setup will use the following system settings:

         DOS Type:        MS-DOS
         MS-DOS Path:     C:\DOS
         Display Type:    VGA

         The settings are correct.

        If all the settings are correct, press ENTER.

        To change a setting, press the UP ARROW or DOWN ARROW key until
        the setting is selected. Then press ENTER to see alternatives.

ENTER=Continue  F1=Help  F3=Exit
```

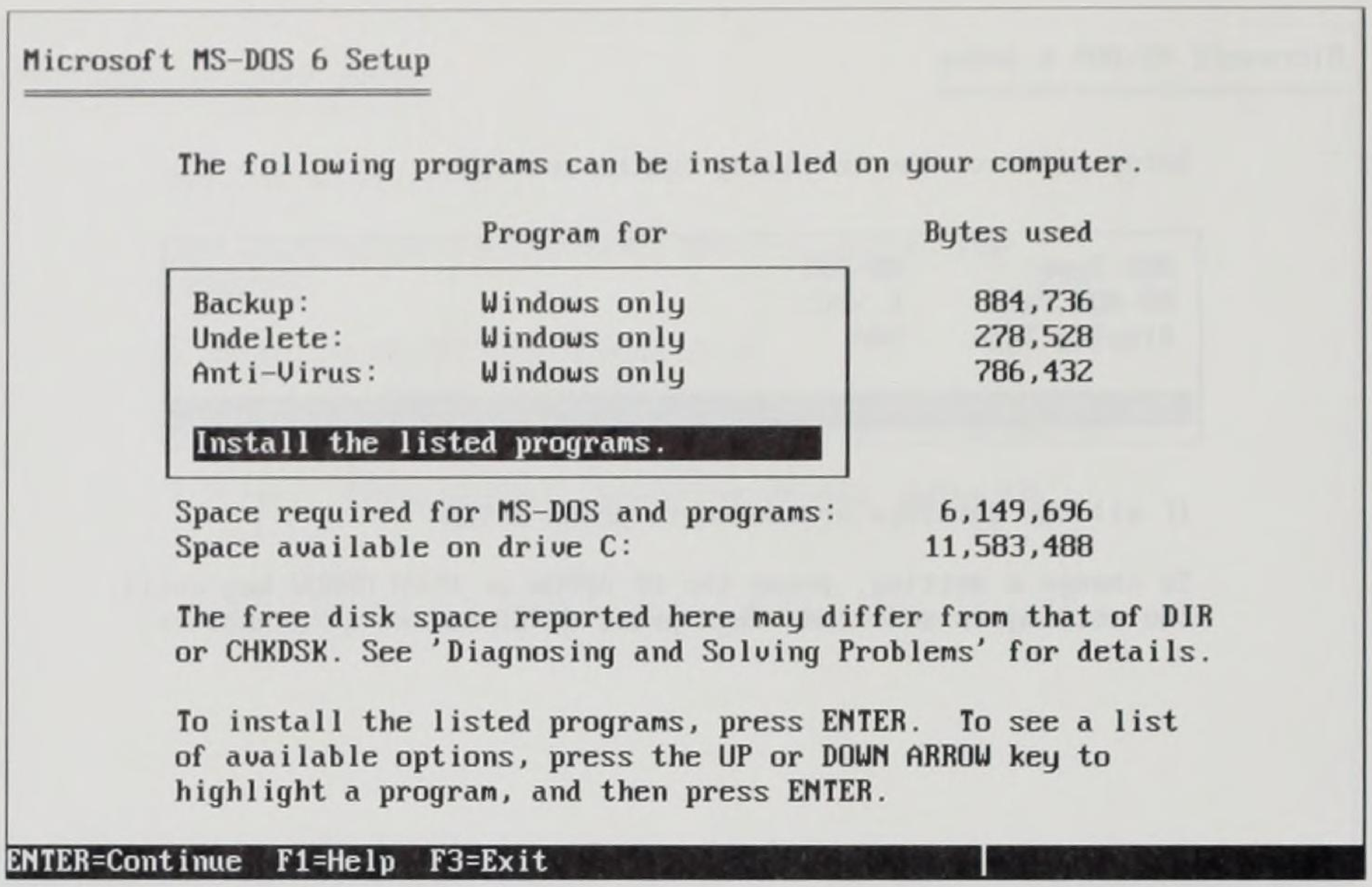
Figure D-3. *The Setup System Information screen*

If the information displayed is correct, press ENTER to continue the upgrade. If one or more of the items displayed is not correct, use the arrow keys to highlight the entry and press ENTER. Type in the correct setting. When the entries are correct, press ENTER to continue the upgrade. Setup will display the screen, shown in Figure D-4, that lets you direct it to install its backup, undelete, and antivirus utilities for DOS, Windows, or both.

If you use Windows, you will want Setup to install the Windows utilities. If you have sufficient disk space, you will want Setup to install the utilities for Windows and DOS. To control which utilities Setup installs, use the arrow keys to highlight an entry and press ENTER. Setup will display a screen similar to that shown in Figure D-5, which lets you select the desired utilities.

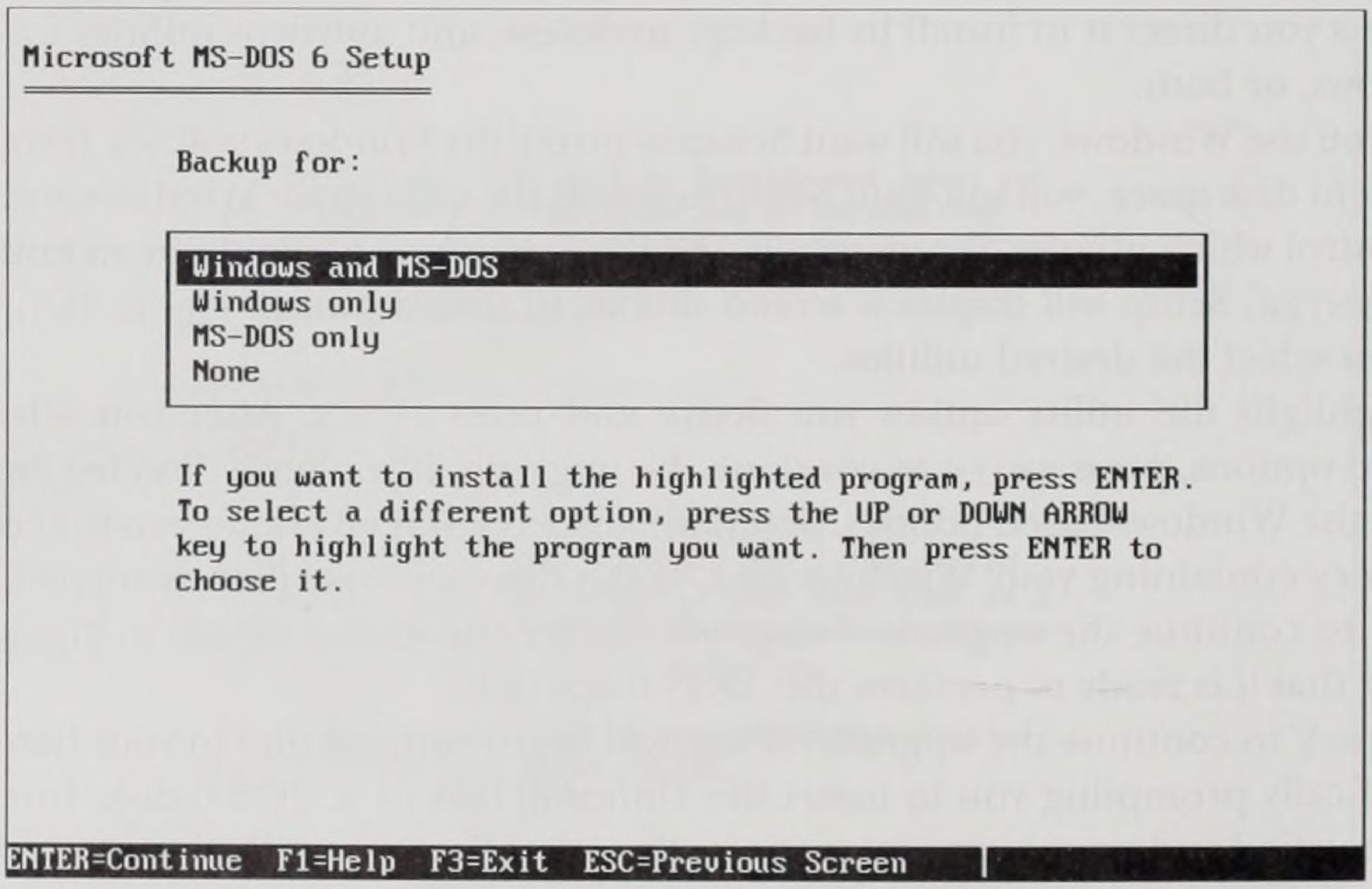
Highlight the utility option you desire and press ENTER. After you select the desired options, press ENTER to continue the upgrade. If you have directed Setup to install the Windows-based utilities, Setup will display a screen asking you to verify the directory containing your Windows files. If the directory specified is correct, press ENTER to continue the upgrade. Setup will display the screen shown in Figure D-6, stating that it is ready to perform the DOS 6 upgrade.

Type **Y** to continue the upgrade. Setup will begin copying files to your hard disk, periodically prompting you to insert the Uninstall disk or a DOS 6 disk. Insert the disk specified and press ENTER to continue the upgrade. Eventually, Setup will display a screen directing you to remove all floppies and to press ENTER. Setup will then display a screen stating that the upgrade is complete and that it has saved to the



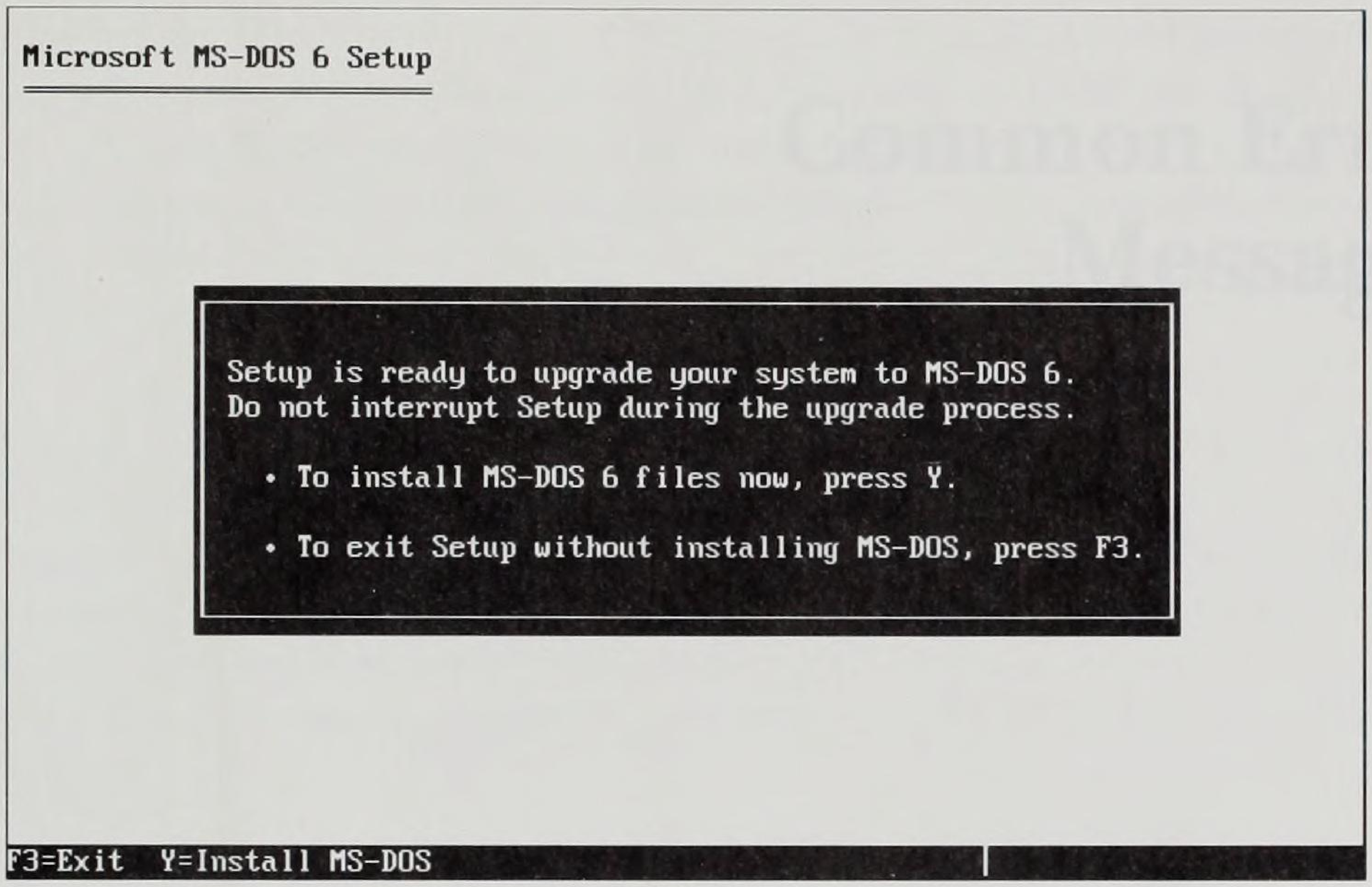
```
Microsoft MS-DOS 6 Setup

        The following programs can be installed on your computer.

                          Program for                Bytes used

         Backup:          Windows only                 884,736
         Undelete:        Windows only                 278,528
         Anti-Virus:      Windows only                 786,432

         Install the listed programs.

        Space required for MS-DOS and programs:       6,149,696
        Space available on drive C:                  11,583,488

        The free disk space reported here may differ from that of DIR
        or CHKDSK. See 'Diagnosing and Solving Problems' for details.

        To install the listed programs, press ENTER.  To see a list
        of available options, press the UP or DOWN ARROW key to
        highlight a program, and then press ENTER.

ENTER=Continue  F1=Help  F3=Exit
```

Figure D-4. *The Setup Utility screen*



```
Microsoft MS-DOS 6 Setup

        Backup for:

         Windows and MS-DOS
         Windows only
         MS-DOS only
         None

        If you want to install the highlighted program, press ENTER.
        To select a different option, press the UP or DOWN ARROW
        key to highlight the program you want. Then press ENTER to
        choose it.

ENTER=Continue  F1=Help  F3=Exit  ESC=Previous Screen
```

Figure D-5. *The Setup Utility Selection screen*



```
Microsoft MS-DOS 6 Setup

Setup is ready to upgrade your system to MS-DOS 6.
Do not interrupt Setup during the upgrade process.

  • To install MS-DOS 6 files now, press Y.

  • To exit Setup without installing MS-DOS, press F3.

F3=Exit  Y=Install MS-DOS
```

Figure D-6. *The Setup Ready to Upgrade screen*

Uninstall disks your AUTOEXEC.BAT and CONFIG.SYS files as files with the DAT extension. To restart your computer in DOS 6, press ENTER.

After your system starts and you verify that it is running correctly, you can remove the DOS files that corresponded to your previous DOS version by issuing the DELOLDOS command as shown here:

```
C:\> DELOLDOS <ENTER>
```

Respond to the prompts DELOLDOS displays on your screen.

Using the Uninstall Disks

The Uninstall disks let you restore your system to its state immediately before you began the DOS 6 installation. Should you experience a fatal error during the DOS 6 upgrade, place the disk labelled Uninstall 1 in drive A and restart your system. Follow the messages that appear on your screen to restore your previous version of DOS.

Appendix E

Common Error Messages

The Command Reference section of this book provides a detailed description of the error messages you may encounter while working with a specific command. This appendix discusses error messages common to many commands.

Abort, Retry, Fail?

DOS has encountered an error it cannot resolve without user intervention, such as an open floppy disk drive latch or an offline printer. To end the command generating this error message, type **A** for Abort. If you can correct the cause of the error, for example, closing the floppy disk drive latch, do so and type **R** to retry the command. The Fail option, the least often used, directs DOS to fail the operation that caused the error and continue the program's execution. If you are copying files from a disk and DOS encounters a disk read error, you may be able to use Fail to fail the reading of the bad sector and continue the file copy with the next file.

Access denied

The command attempted to delete or change a read-only file or specified a filename as the desired directory in a CHDIR command. DOS does not let you change read-only files. If you really want to change or delete the file's contents, use ATTRIB to remove the file's read-only setting.

Bad command or file name

DOS could not locate the command specified as an internal or external command or DOS batch file. Double-check the command's spelling. Make sure the directory containing the command is in the command path (see PATH) or that you are specifying a complete directory path name to the command.

Bad or missing Command interpreter

DOS could not find the file COMMAND.COM during system startup. By default, DOS looks in the root directory of the boot disk for the command interpreter file COMMAND.COM. If your CONFIG.SYS file contains a SHELL entry, you can direct DOS to look in a location other than the root. If this error occurs, boot DOS from a floppy disk and either move COMMAND.COM to the root directory or place a SHELL entry in CONFIG.SYS that contains a complete path to COMMAND.COM.

Batch file missing

DOS cannot continue the execution of the current batch file because it cannot find the batch file in the current directory or command path. This error message may occur if a batch file deletes itself as part of its processing.

Cannot load COMMAND, system halted

The previous command overwrote the transient portion of COMMAND.COM in memory. When this occurs, DOS uses the COMSPEC environment entry to locate the file COMMAND.COM to reload the transient portion. In this case, DOS was unable to find or load COMMAND.COM. Use CTRL-ALT-DEL to restart your system.

Cannot read file allocation table

See "File allocation table bad."

Current drive is no longer valid

The current drive is either a floppy drive with no disk in it or a network drive that is no longer accessible. You must either change the current drive or, in the case of a floppy drive, insert a disk.

Data error reading drive *n*:
Disk error reading drive *n*:

Either of these error messages means DOS cannot read the data stored on the disk drive specified. DOS will follow this message with the "Abort, Retry, Fail" prompt. Use the Fail option to read as much of the data from the disk as you can. If you cannot read any information, you might try using the RECOVER command. Then, use FORMAT to reformat the disk.

Disk error writing drive *n*:

DOS cannot write information to the disk drive specified. DOS will follow this message with the "Abort, Retry, Fail" prompt. Abort the operation trying to write information to the disk. Use CHKDSK to examine the disk. If possible, salvage as much data from the disk as you can using COPY and RECOVER. Then, use FORMAT to reformat the disk.

Divide overflow

The command performed an invalid division operation, normally a divide by zero. Restart your system using CTRL-ALT-DEL and determine if the error persists. If so, contact the manufacturer of the software program for assistance.

Error writing directory

DOS cannot write to the directory specified. Restart your system and use CHKDSK to examine your disk. Copy as much information from the disk as you can, then use FORMAT to reformat the disk.

Error writing file allocation table

DOS cannot write to the file allocation table specified. DOS uses the file allocation table to track the location of every file on your disk. To keep a file allocation table error from preventing DOS accessing your files, DOS stores two copies of the file allocation table on most disks. Use BACKUP or COPY to copy information from the disk, then use FORMAT to reformat the disk.

File allocation table bad
File allocation table bad drive *n*:
Cannot read file allocation table

Any of these three error messages means the file allocation table for the disk specified may be damaged. Use CHKDSK to examine the disk. DOS uses the file allocation table to track the location of every file on your disk. To keep a file allocation table error from preventing DOS accessing your files, DOS stores two copies of the file allocation table on most disks. If one of the file allocation tables becomes damaged, DOS uses the second. Use COPY or BACKUP to copy as many files from your disk as you can. Use FORMAT to reformat the disk.

File creation error

The command attempted to create a file whose name is already in use as a directory name within the same directory, or the command attempted to create a file in the root directory and the root is full.

File not found

The command could not locate the file you specified in the command line. Double-check your spelling of the filename. Make sure you are specifying a correct path to the file. Use the DIR command to ensure that the file is in the directory you are referencing.

General failure error reading (or writing) drive *n*:

DOS cannot access the disk drive specified. This message is followed by an "Abort, Retry, or Fail" prompt. Select the Abort option. In most cases, this error occurs when the disk in the drive specified has not been formatted.

Incorrect DOS Version

The command or program you are trying to execute requires a different DOS version than you are using. Locate either the correct DOS version or the program version that runs under your version of DOS. If you are using DOS 5 or later, you may be able to execute the program using SETVER.

Incorrect number of parameters
Invalid number of parameters
Too many parameters

Any of these three error messages means that the command line you entered has too many or too few parameters. Examine the command's format in the Command Reference section of this book and then issue the command with the correct parameter count.

Insert disk with batch file and press any key when ready

The floppy disk from which you originally started the batch file is no longer in the current drive. To continue the batch file's execution, place the disk containing the batch file back into the drive and press any key.

Insert disk with \COMMAND.COM in drive *n*: and strike any key when ready

The previous command overwrote the transient portion of COMMAND.COM in memory. When this occurs, DOS uses the COMSPEC environment entry to locate the file COMMAND.COM to reload the transient portion. In this case, DOS is prompting you to insert a floppy disk into the drive specified that contains COMMAND.COM, so DOS can reload the transient portion.

Insufficient disk space

The disk specified is full, preventing the previous command from successfully finishing. If possible, free up space on the disk by removing unneeded files or copying one or more files to a different disk. If you are trying to copy files from one disk to another, see the XCOPY command in the Command Reference section of this book.

Insufficient memory
Not enough memory

Either of these error messages means that your computer does not have enough available memory for the command to execute. If you have loaded memory-resident programs or a RAM drive, you may need to restart DOS without them to successfully execute this command. If you are using DOS 5 or later, refer to the CONFIG.SYS DEVICEHIGH entry and the LOADHIGH command for ways to free up conventional memory.

Intermediate file error during pipe

The DOS pipe creates temporary files to which it redirects a program's output. If the disk is write-protected or its root directory is full, DOS cannot create the needed files. If you are using DOS 5 or later, you can use the TEMP environment entry to specify where you want DOS to create the pipe's temporary files.

Internal stack overflow, system halted

A series of successive hardware interrupts used up all of the DOS interrupt stack space. Use the CONFIG.SYS STACKS entry to allocate more stack space and restart your system.

Invalid Argument
Invalid Parameter

Either of these error messages means that one or more parameters you specified in the command is invalid. Refer to the command in this book's Command Reference, noting the valid parameters, and reissue the command. If you are using DOS 5 or later, invoke the command with the /? switch to display a list of valid parameters.

Invalid COMMAND.COM in drive *n*:

The previous command overwrote the transient portion of COMMAND.COM in memory. When this occurs, DOS uses the COMSPEC environment entry to locate the file COMMAND.COM to reload the transient portion. In this case, when DOS reads COMMAND.COM from the disk specified, the version of COMMAND.COM on disk did not match the version DOS is currently running. Insert a disk containing the correct version of COMMAND.COM and press any key to continue.

Invalid drive in search path

A disk drive in either the command-file search path defined by PATH or the data-file search path defined by APPEND does not exist. Invoke either PATH or APPEND with no arguments to view their current settings and remove or correct the invalid disk reference.

Invalid drive or file name

A drive letter or filename specified in the command line is invalid. Make sure you are specifying a valid disk drive letter, followed by a colon, or that you are providing a correct path name to the file specified.

Invalid drive specification

The disk drive specified in your command line is invalid. Make sure you are using a valid disk drive letter followed by a colon. If you are trying to access your hard disk for the first time, the disk may not yet have been partitioned by FDISK.

Invalid number of parameters

See "Incorrect number of parameters."

Invalid Parameter

See "Invalid Argument."

Invalid parameter combination
Parameters not compatible

Either of these error messages means that your command line contains two or more conflicting switches. Refer to the command in this book's Command Reference, noting the valid parameters, and reissue the command. If you are using DOS 5 or later, invoke the command with the /? switch to display a list of valid parameters.

Invalid path
Path not found

Either of these error messages means that the directory path you have specified in your command does not exist. Double-check your spelling and make sure you are providing a correct path name relative to your current directory. Reissue the command with a correct path name.

Invalid switch

One or more of the switches you have specified in the command line is not supported by this command. Refer to the command in this book's Command Reference, noting the valid switches, and reissue the command. If you are using DOS 5 or later, invoke the command with the /? switch to display a list of valid parameters.

Invalid syntax

The command line entered violates the command's format. If you are using the FOR command, make sure you include the keyword DO. If you are using an IF command within a batch file to compare one string to a batch parameter, enclose both strings within single quotes to ensure the IF command has two strings to compare if the user does not provide a corresponding batch line parameter.

No such file or directory

A file or directory specified in the command line does not exist. Double-check your spelling of the file or directory name. Make sure you are specifying a correct path name. Use DIR to ensure that the file or directory is in the directory you are referencing.

Not enough memory

See "Insufficient memory."

Not ready error reading (or writing) drive *n*:

The command attempted to read information from or write information to the disk in the drive specified, and the drive was not ready. This error occurs when you don't have a floppy disk in the drive specified or you have not closed the disk drive latch. This message is followed by the "Abort, Retry, or Fail" message. If you want to end the command trying to access the disk, select Abort. If you have inserted the disk or closed the disk drive latch, select Retry.

Parameter value not allowed
Parameter value not in allowed range

Either of these error messages means that the value you have specified for one of the parameters in the command line is invalid. Refer to the command in the Command Reference section of this book to determine the valid range of values.

Parameters not compatible

See "Invalid parameter combination."

Path not found

See "Invalid path."

Path too long

The path specified exceeds the DOS maximum of 64 characters.

Program too big to fit in memory

You do not have enough available memory to run the program specified. If you have loaded memory-resident programs or created a RAM drive in your computer's conventional memory, start your system without loading them into memory.

Out of environment space

The command could not place an entry in the DOS environment because the environment is full and DOS cannot increase its size due to memory-resident programs. Use the CONFIG.SYS SHELL entry to increase the amount of space DOS reserves for the environment and restart your system.

Sector not found reading (or writing) drive *n*:

DOS could not find the sector corresponding to a disk read or write operation on the drive specified. Use CHKDSK to examine the disk. Restart your computer and retry the program. If the error persists, perform a backup, format, and restore operation.

Seek error reading (or writing) drive *n*:

A seek operation moves the disk drive heads to a specific track. A seek error occurs when the drive is unable to seek to the track specified. Use CHKDSK to examine the disk. Restart your computer and retry the program. If the error persists, perform a backup, format, and restore operation.

Specified COMMAND search directory bad

The CONFIG.SYS SHELL entry lets you specify the location of the DOS command interpreter COMMAND.COM. In this case, the location specified does not exist or does not contain COMMAND.COM. If necessary, restart your system using a bootable floppy disk in drive A, and correct the CONFIG.SYS SHELL entry.

Terminate batch job (Y/N)?

You have pressed either CTRL-BREAK or CTRL-C during the execution of a DOS batch file. If you type **Y**, DOS will end the batch file's execution, returning control to the

DOS prompt. If you instead type N, DOS will end only the current command's execution, continuing the batch file with the next command.

Too many open files

The command could not execute because DOS did not have enough file handles available. Use the CONFIG.SYS FILES entry to increase the number of available file handles and restart your system. At a minimum, your system should provide 20 file handles.

Too many parameters

See "Incorrect number of parameters."

Write protect error writing drive *n*:

The command attempted to write information to or delete files from a write-protected disk. If you want to change the contents of the disk, you must first remove the write-protect tab from a 5 1/4-inch floppy disk or move the write-protect bar on a 3 1/2-inch disk. If you don't want to change the disk's contents, make sure you are specifying the command's source and target disks in the correct order.

Index

* (Asterisk wildcard), 153-161
. (Directory abbreviation), 203-205, 208
.. (Directory abbreviation), 203-205, 208
< (Input redirection operator), 352-353, 357
> (Output redirection operator), 347-350, 357
>> (Append redirection operator), 350-351, 357
? (Question mark wildcard), 161-163, 164
| (Pipe operator), 367-373

A

Abort, 116
Abort, Retry, Fail? message, 108-109, 110, 1089
Access denied message, 1089
Access time, 829
Advanced power management, 794-795, 797
Allocation unit (*See also* cluster), 131-134, 135, 478-480, 489
ANSI.SCR debug script file to create ANSI.COM (aide to writing sequences), 670-671
ANSI.SYS device driver, 671
  /K switch, 669
  /X switch, 669
  assigning text attributes with, 676-677
    escape sequences, table of, 677
  controlling screen colors with, 663-664
    background color values, table of, 663
    foreground color values, table of, 664
  cursor positioning with, 673-675
    escape sequences, table of, 675
  installing, 659
  redefining keys with, 665-669
  selecting a video mode with, 675-676
    escape sequences, table of, 676
  setting screen size with, 676-677
APM (advanced power management), 794-795
APPEND command, 212-213, 865-868
  /E switch, 866
  /PATH switch, 866
  /X switch, 866
  messages, 867
Append redirection operator (>>), 350-351, 357
Archive bit, 390
ASCII character set, 1065-1068
ASCII files, 138
ASSIGN command, 491-493, 497-498, 867-868
  /STATUS switch, 493, 868

    - commands not to use with, table of, 493
- Asterisk wildcard (*), 153-161
    - COPY command with, 157-159
    - DEL command with, 160-161
    - DIR command with, 153-157
    - RENAME command with, 159-160
- ATTRIB command, 226, 417-421, 869-870
    - +/- A switch, 869
    - +/- H switch, 869
    - +/- R switch, 869
    - +/- S switch, 869
    - /S switch, 869
    - messages, 870
- Attribute byte, 390
- AUTOEXEC.BAT file, 41, 321-331
    - cautions when changing, 327-328
    - common commands, 322-326
        - table of, 322
    - CONFIG.SYS and, 327
    - DIR command attributes, defining in, 325
    - disabling, 331
    - DOS Shell, invoking in, 325
    - FASTOPEN command in, 323-324
    - Microsoft Windows, invoking in, 326
    - PATH command in, 322-323
    - PRINT command in, 324
    - PROMPT command in, 323
    - sample file, 326-327
    - SHARE command in, 326
    - shielding from software installations, 330
    - SMARTDRV command in, 326
    - temporary files, defining location of in, 324-325
    - VSAFE command in, 326

B

- Backplane, 29
- BACKUP command, 386-392, 871-874
    - /A switch, 390, 871
    - /D switch, 410, 871
    - /F switch, 386, 871
    - /L switch, 393-394, 871
    - /M switch, 390, 871
    - /S switch, 387, 390, 871
    - /T switch, 410, 871
    - exit status values, 413, 872, 1059
    - large files and, 412
    - log file, 393-394
    - messages, 873-874
    - using in batch files, 413-414
- Backup
    - backing up a key file or directory, 391
    - backing up entire hard disk, 386-388
        - determining the number of floppy disks required, 384-385
    - complete, 395
    - differential, 397
    - full, 397
    - incremental, 395, 397
    - performing daily backup operations, 388-391
    - understanding the process, 384-391
- Bad command or file name message, 52-53, 1090
- Bad or missing Command interpreter message, 1090
- Bad sector, 135
- Base name, 65, 83
- BASIC programming language, invoking, 785-786
- Batch file, 259, 268, 277
    - building on the fly, 362-365
    - disabling command-name display in, 262-263
    - ending, 263-264, 269
    - invoking one from another, 311-313

labels, 304
  length of, 304-305
labels, locating with PATH, 264-265
named parameters in, 294-296
naming, 262
passing information to, 287-291
placing remarks in, 266
temporarily suspending, 266-268

Batch file missing message, 1090
Batch parameters, 287, 292, 301
  accessing more than nine, 308-311
Baud rate, 814
BEEP.COM program to sound the built-in speaker one time, 557
BELLS.BAT batch file to sound the computer's speaker three times, 276
Block device, 460
BLUEBACK.BAT batch file to select a blue screen background, 660
BOOT_DOS.SCR debug script file that creates program to reboot DOS, 562-563, 565
Boot record, 128
  entries included in, 488
Branch, 320
BREAK command, 778-779, 796, 874-875
  in CONFIG.SYS, 456
  messages, 875
  system performance and, 607
Buffer, 472
BUFFERS CONFIG.SYS entry and system performance, 602-603
Bug, 553, 582
Bus, 30
Byte, 13, 19, 30

C

CALL command, 311-313, 876
Cannot load COMMAND, system halted message, 1090
Cannot read file allocation table message, 1090, 1092
Central processing unit (CPU), 2
Character device, 460
Character set, 715
Character string, 287
Chassis, function of, 2
  illustration of common components in, 25, 26
CHCP command, 726-727, 729, 876-878
  code-page values, 877
  messages, 877-878
CHDIR (CD) command, 173-176, 182, 878-879
  messages, 879
CHECK1ST.BAT batch file that copies specific files to floppy disk, 283
CHECKFOR.BAT batch file that checks the root directory for specific files, 282-283
CHKDCOPY.BAT batch file that checks the success of a DISKCOPY operation, 286
CHKDSK command, 473-485, 879-884
  /F switch, 480-481, 879, 880
  /V switch, 484, 879
  correcting disk errors with, 480-484
  detecting fragmented files with, 819-820
  displaying filenames with, 484
  messages, 881-884
  output from, 475-480
  rebuilding cross-linked files with, 488-489
CHKDSK, operation of, 654-655
CHOICE command, 297-300, 885-886
  /C switch, 885
  /N switch, 885
  /S switch, 885
  /T switch, 885
  case-sensitive processing, 299-300
  messages, 886
  specifying default option, 299

- suppressing display of valid keys, 300
- Client, 689, 696, 698
- Clock tick, 255
- CLS command, 44, 57, 887
- Clusters, 489, 683
  - number of commonly available, 648
- Code page, 715, 731
  - preparing, 723-724
  - selecting, 724-725
  - support files, table of, 723
  - switching, 731
- Code Page Tables, 1076-1081
  - changing, 726-727
  - displaying status information, 725
  - refreshing, 725-726
  - table of supported, 717
- Cold boot, 93, 101, 103
- COLOR.BAT batch file that checks the value of an argument, 288-289
- COLORS.BAT batch file that checks the value of an argument, 316, 317
- COM (communications port), 231
- COMMAND command, 887-889
  - /C switch, 311-313, 888
  - /E switch, 888
  - /K switch, 888
  - /MSG switch, 888
  - /P switch, 331, 888
  - messages, 889
- COMMAND.COM program, 41
- Command line, 50, 61
- Command-line editing, advanced, 99-100
- Command path, 215
- Command processor, customizing, 460-462
- Commands, installing memory resident, 458-459
- COMP command, 424-429, 435, 889-893
  - /A switch, 427, 890
  - /C switch, 428, 890
  - /D switch, 890
  - /L switch, 427, 890
  - /N switch, 429, 890
  - messages, 892-893
- Complete disk backup, 416
- Compressed file, 441
- Computer program, 17
- CONFIG.SYS file, 41, 331
  - AUTOEXEC.BAT and, 327
  - BREAK entry, 456, 463
  - BUFFERS entry, 447-449, 453
    - /X switch, 448
    - default settings, table of, 448
    - recommended settings, table of, 447
  - COUNTRY entry, 450-451
  - device drivers, table of, 452
  - DEVICE entry, 451-452, 453
  - DOS 6 and, 464-472
  - DOS 6 entries, table of, 465
  - DRIVPARM entry, 460, 464
    - form factor values, table of, 461
  - entries, table of, 445
  - FCBS entry, 457, 463
  - FILES entry, 450, 453
  - INCLUDE entry, 469-470
  - INSTALL entry, 459, 463-464
  - LASTDRIVE entry, 456-457, 463
  - MENUCOLOR entry, 467-468
    - color values, table of, 468
  - MENUDEFAULT entry, 468-469
  - MENUITEM entry, 466-467
  - REM entry, 451, 453
  - SET command, 472
  - SHELL entry, 460-462, 464
    - /E switch, 345
  - STACKS entry, 458, 463
    - default values, table of, 458
  - SUBMENU entry, 470-471
  - SWITCHES entry, 459-460
- Contiguous file, 816, 830
  - illustration of, 816

Control-Break checking, enabling extended, 778-779
Controller overhead, 829
Conventional memory, 613, 633, 644
COPY command, 139-143, 146, 894-896
        /A switch, 894
        /B switch, 894
        /V switch, 894
        messages, 896
Copying files,
        floppy disk to hard disk, 218-219
        hard disk to floppy disk, 217-218
COPYMSG.BAT batch file example of copying files without output redirection, 362
Country code values, table of, 719
COUNTRY CONFIG.SYS entry, 718-720, 727
CP/M, 857
CPU types, table of, 29
CRC (cyclic redundancy check), 687
Cross-linked files, 482, 489
        rebuilding with CHKDSK command, 488-489
CTRL-ALT-DEL, restarting DOS with, 93-94, 95
CTRL-C, ending a command with, 92-93, 95
CTRL-PRTSC (printer echo), 236, 237
CRTL-S, suspending output with, 102
CTTY command, 801, 896-897
        messages, 897
Current directory, 172-173, 191
        saving and restoring, 319-320
Current drive is no longer valid message, 1090
Cursor, positioning with ANSI.SYS, 673-675
Cut and paste, 714
Cyclic redundancy check, 687
Cylinders, 118

D

DAILY.BAT batch file to perform daily backup operations using BACKUP command, 413
Data error reading drive *n*: message, 1091
Data-file search path, 215
Data-transfer time, 829
DATE command, 49-50, 61, 897-898
        messages, 898
Date stamp, 68, 74, 83
DATETIME.BAT batch file to invoke DATE and TIME commands, 260-261
DBLSPACE command, 845-866, 899-902
        /CHKDSK switch, 854, 899
        /COMPRESS switch, 899
        /CONVSTAC switch, 899
        /CREATE switch, 899
        /DEFRAG switch, 854, 900
        /DELETE switch, 900
        /FORMAT switch, 900
        /INFO switch, 900
        /LIST switch, 853, 900
        /MOUNT switch, 900
        /RATIO switch, 900
        /SIZE switch, 900
        /UNMOUNT switch, 900
        compressed disk volume, 848, 850
                checking status of, 854
                defragmenting, 854
        floppy disks and, 853
                mounting and unmounting, 853-854
        setting up, 845-848
        SMARTDRV command and, 850
        uncompressed disk volume, 848, 850
                files residing on, 848
DBLSPACE.SYS device driver, 849-850, 851
DEBUG, 553, 582
        creating commands with, 553-557
DEBUG commands, 565-582

table of, 566
?, 565-567
A, 567-568
C, 568-569
D, 569
E, 570
F, 570-571
G, 571-572
H, 572
I, 572-573
L, 573
M, 573-574
N, 574
O, 574-575
P, 575
Q, 575-576
R, 576-577
S, 577-578
T, 578-579
U, 579
W, 579-580
XA, 580-581
XD, 581
XM, 581
XS, 582
Debug script file, 557-558, 582
Debugger, 582
Default devices assigned by DOS, 360
names, table of, 361
Default drive, 116
changing, 105-106
DEFINED.BAT batch file determines if batch parameters %1 through %3 are defined, 294
DEFRAG command, 822-826, 902-904
/B switch, 902
/F switch, 902
/S switch, 903
/SKIPHIGH switch, 903
/U switch, 903
/V switch, 903
exit status values, 903-904, 1059-1060
FASTOPEN command and, 823
removing unused space between files with, 824-825
sorting directories with, 824-825
DEL command, 144-145, 146, 904-906
/P switch, 149-150, 905
messages, 906
DELETEIT.BAT batch file used with SELERASE.BAT, 363
Deletion tracking, 552
DELTREE command, 214-215, 906-907
/Y switch, 214, 906
messages, 907
Destination file, 152
Device driver, displaying information about with MSD command, 813
DEVICEHIGH CONFIG.SYS entry, 635
Dialog box, 703, 714
Differential backup, 416
DIP switch, 240
illustration of, 240
DIR command, 64-65, 908-911
/A switch, 80-81, 908, 909-910
/B switch, 82, 908
/L switch, 81-82, 908
/O switch, 79-80, 908, 909
/P switch, 70-71, 908
/S switch, 81, 908
/W switch, 72-73, 908
attribute options, 80-81
table of, 80
command line switches, table of, 338
displaying a wide listing, 72-73
listing for a specific file, 71-72
messages, 911
output, illustration of, 65
sort options, table of, 79
sort order, 79-80
DIR1LINE.BAT batch file that displays a directory, 316
DIRCMD environment entry, defining in AUTOEXEC.BAT, 325

Direct memory access (DMA), 644
Directory, 63, 74, 83
    avoiding fragmented, 187
    current, 172-173
        tracking with PROMPT command, 172-173
    maximum number of files in, 600-601
    maximum performance and, 599-601
    number of files in, 187-188
    protecting private, 188-190
Directory entry, 649-651, 655
    illustration of, 651
    removing unused entries, 612
    unused entries and system performance, 610-612
Directory names, abbreviating, 202-203, 207-208
Directory tree, 178, 191, 215
    deleting an entire, 214
    displaying, 193-197
    illustration of, 178, 194
DIRSONLY.BAT batch file, 709-710
Disk
    accessing without changing default, 106-108, 110
    changing default, 105-106, 109
    common drive configurations and names, illustration of, 1
    drive names, 15-16
    maximum performance and, 602-603
    recording information on, 19
Disk buffer, 444
    improving performance with, 444-449
Disk cache, 603, 612
    creating, 603-606
Disk controller, 26-27, 114-115, 116
    types of, 27
Disk drive specifier, 116
Disk drives, displaying information about with MSD command, 811
Disk error reading (or writing) drive *n*: message, 1091
Disk formatting, 135
Disk head crash, 686, 688
Disk head parking, 679-682, 686, 688
Disk head seek, 830
Disk partitions, 39, 42, 831-833
    illustration of, 39
Disk sector interleaving, 682-684, 686
Disk space, determining available, 68
Disk verification, 598
Disk volume label, 583
    assigning, 585-586
    displaying, 586-587
Disk, floppy, 2
    caring for, 12-13
    disk activation light, 15, 19
    drive spindle, 11
    envelope, 9
    hub, 10
    illustration of, 8, 15
    index hole, 10
    inserting into drive, 14
        illustration of, 16
    jacket, 9
    read/write opening, 9
    sector notch, 11
    shutter, 11
    sizes of, 14
    storage capacity, table of, 14
    write-protect notch, 10, 11
Disk, hard, 2
    cutaway drawing of, 7
    formatting, 123-124
DISKCOMP command, 588-589, 593-594, 911-915
    /1 switch, 912
    /8 switch, 912
    exit status values, 912, 1060
    messages, 913-915

DISKCOPY command, 219-223, 224, 915-919
  /1 switch, 915
  /V switch, 915
  exit status values, 284, 916, 1060
  messages, 917-919
  using one drive, 221-223
  using two drives, 219-221
DISPLAY.SYS device driver, 717, 727
Divide overflow message, 1091
DOS, 19
DOS 1, 858
DOS 1.1, 859
  commands provided with, table of, 858
DOS 2, 859-860
  commands introduced with, table of, 860
DOS 3, 860-861
  commands introduced with, table of, 860
DOS 4, 861
DOS 5, 861-862
  commands introduced with, table of, 862
DOS 6, 862-863
  commands introduced with, table of, 863
DOS, starting, 37
  from floppy disk, 31
DOS booting, 42
DOS boot record, 40
  illustration of, 40
DOS commands, executing, 43
DOS prompt, 33, 42
DOS Shell, 733-758
  associating program with file type, 740-741
  backing up a disk, 754
  changing file attributes, 746-747
  changing the number and order of files displayed, 748
  confirmation boxes, enabling and disabling, 749
  confirmation options, table of, 749
  copying files, 744-745
  copying a disk, 754
  creating a directory, 747-748
  deleting files or directories, 745-746
  dialog box, designing, 772-773
  directory tree, 736
    collapsing a directory's branches, 737-738
    expanding a directory tree, 737
    expanding the next level of directories, 737
    working with, 737
  disk drive icons, illustration of, 735
  display, changing, 750-752
  displaying file specifics, 749-750
  displaying in graphics mode, 737
  DOSSHELL.INI, 773
  exiting from, 738
  file list, 736
  file menu options, table of, 765
  file type icons, illustration of, 736
  formatting a disk, 755
  help menu options, table of, 757
  hot key, 734-735
  interface, illustration of, 34, 35, 734
  invoking in AUTOEXEC.BAT, 325
  key combinations, table of reserved, 768
  main group, 736
  menu bar, 734
  mouse, using to move or copy files, 745
  moving files, 742-744
  printing files, 740
  program entries, 767-769
    adding new, 767-769

        changing, 769
        copying to a new group, 769
        editing existing, 766
    program group, 764, 770, 774
        adding new, 764-766
        changing the order of, 767
        removing, 766-767
    renaming files, 746
    replaceable parameter, 774
    restoring a disk, 755
    running programs, 739-740
        from the program list, 752
    searching for specific files, 741
    selecting files, 739
        from different directories, 739
    selecting menu option, 734
    setting screen colors in, 763-764, 770
    task swapper, 760-762
    title bar, 734
    undeleting files, 755-756
    using online help, 756-757
    viewing file's contents, 742
DOS startup, troubleshooting, 36
DOS= CONFIG.SYS entry, 627-630, 635
DOSKEY command, 919-922
    /BUFSIZE switch, 512, 919
    /HISTORY switch, 509, 919
    /INSERT switch, 512, 919
    /MACROS switch, 919
    /OVERSTRIKE switch, 919
    /REINSTALL switch, 512, 919
    command editing keys, 920
    discarding buffer contents with ALT-F7 key, 510
    displaying command history with F7 key, 509
    editing keys, table of, 511
    editing previous commands with, 511
    function keys, table of, 510
    messages, 922
    metacharacters, 920-921
    recalling commands with, 507-510
    recalling commands with F8 key, 510
    selecting line number with F9 key, 509
    traverse keys, table of, 508
DOSKEY macros, 514-520
    accessing command-line parameters from, 516-517
    function of, 519-520
    I/O redirection in, 518-519
    I/O redirection symbols, table of, 518
    restrictions, 519
    special symbols supported by, table of, 517
DOSSHELL command, 922-923
    /B switch, 922
    /G switch, 922
    /T switch, 922
DOWP.BAT batch file to invoke WordPerfect, 261-262

E

ECHO command, 265, 269, 923-925
    displaying blank lines with, 274-275, 924
    using CTRL-G to create sound with, 276
    using extended ASCII characters with, 275-276
ECHO OFF command, 265, 268
EDIT command, 925-926
    /G switch, 925
    /H switch, 925
    /NOHI switch, 925
    change dialog box, illustration of, 712
    clear option, 713
    copy option, 713
    creating a new file, 707-708
    cursor movement commands, 705-706

- cursor movement key combinations, illustration of, 706
- customizing the screen display, 708-709
- cut and paste operations, 713
- cut option, 713
- display options dialog box, 708-709
  - illustration of, 709
- editing an existing file, 702, 706-707
- empty editing window, illustration of, 700
- exiting without saving changes, 702-703
- file menu, 706-707
  - illustration of, 701
- file menu save as option, 708
  - illustration of, 708
- file open dialog box, illustration of, 707
- find text dialog box, illustration of, 712
- online help, 704-705
  - key combinations, table of, 705
  - menu, illustration of, 704
  - screen, illustration of, 704
- paste option, 713
- printing file, 703
- search menu, illustration of, 711
- search operations, 711-713
- starting, 699-700

EMM386.EXE driver, 642-644
- /P switch, 642-643
- B parameter, 643
- D parameter, 644
- FRAME parameter, 642-643
- H parameter, 644
- I parameter, 643
- L parameter, 643
- M parameter, 642-643
- W parameter, 642
- X parameter, 643

Environment, 333-346
- adding entries, 334-335, 339
- COMSPEC entry, 343-345
- DIRCMD entry, 336-337
- DOS use of, 336-337

Environment entry, 346
- form of, 334
- increasing size during system startup, 342-343, 462
- removing, 335-336, 339
- TEMP entry, 336-337

Error messages, 1089-1098

Error messages, loading into memory with CONFIG.SYS SHELL entry, 462

Error writing directory message, 1091

Error writing file allocation table message, 1091

ESC, canceling a command with, 100-101, 102

Escape character, 660

Escape sequence, 660, 677
- creating with EDIT, 662
- creating with EDLIN, 661-662
- creating with a word processor, 662-663

ESDI (Enhanced Small Device Interface) disk controller, 27

EXIT command, 780, 796, 926

Exit status value, 301

Exit status values, tables of, 1059-1063

EXPAND command, 440-441, 926-927

Expanded memory, 615-619, 633-634, 644

Expanded memory specification, 615
- differences in version 4.0, 619
- use of, 621-623

Expansion slot, 27
- illustration of inserting a board into, 28

Expansion slot cover, 27
- illustration of removing, 28

- Extended ASCII character set, 1065-1068
  - displaying with GRAFTABL, 784-785
  - illustration of, 276
- Extended memory, 123-125, 634, 644
  - use of, 621-623
- Extended partition, 832, 844
- Extension, 65-68, 83
  - common DOS, table of, 66
  - common user, table of, 66
  - examples of meaningful, table of, 67
- External commands, 51, 61
  - execution of, 197-198
  - file name extensions, 60
  - table of, 53
- External drive, 115

F

- F1 function key, repeating letters of a command with, 89
- F2 function key, copying characters with, 99, 101
- F3 function key, repeating commands with, 87-89, 94
- F4 function key, copying characters with, 99, 101
- F5 function key
  - editing current command with, 100, 102
  - pressed during system startup, 471
- F6 function key, indicating end-of-file with, 90-91, 95
- F8 function key, pressed during system startup, 471
- F10 TREE.BAT batch file to redefine the F10 key to display the TREE command, 665-666
- FASTHELP command, 782, 927-928
  - messages, 928
- FASTOPEN command, 205-206, 208, 600, 929-931
  - /X switch, 929
  - in AUTOEXEC.BAT, 323-324
  - messages, 930-931
- FAT (*See also* File Allocation Table), 658
- FC command, 429-431, 435, 931-935
  - /A switch, 931
  - /B switch, 931
  - /C switch, 931
  - /L switch, 931
  - /Lb switch, 931
  - /N switch, 931
  - /T switch, 931
  - /W switch, 931
  - command-line switches, table of, 431
  - messages, 934-935
- FDISK command, 834-841, 935-941
  - and master boot record, 843-844
  - creating a primary and an extended partition, 837-841
  - creating one large partition, 835-837
  - deleting a partition, 843
  - messages, 937-941
- File
  - maximum per directory, 600-601
  - maximum performance and, 601-602
- File allocation table (FAT), 129, 135, 647-649, 655, 658
  - location of on disk, 651-653
- File allocation table bad drive *n*: message, 1092
- File allocation table bad message, 1092
- File attributes
  - displaying with ATTRIB command, 418-419, 434
  - setting with ATTRIB command, 419-421, 434
- File control block (FCB), 457, 462, 597-598
  - illustration of, 598
- File creation error message, 1092

File handle, 360, 450, 472
    and CONFIG.SYS FILES entry, 361
File list, 762
File not found message, 1092
Filename, 65-68, 74
    ASCII, 138
    determining number in a directory, 68
    examples of meaningful, table of, 67
    listing in a specific directory, 68-70
    valid characters, 67-68
Filter, 381
FIND command, 941-944
    /C switch, 356, 941
    /I switch, 941
    /N switch, 941
    /V switch, 942
    command line switches, table of, 356
    comparing the contents of two, 588-589
    formatting a disk with less storage capacity than the drive, 123
    messages, 943-944
    preparing for use, 121-123
    redirection and, 355-356
FOR command, 307-308, 313-314, 944-945
    executing from DOS prompt, 308
    messages, 945
    wildcard characters with, 307-308
FORMAT command, 121-124, 945-952
    /1 switch, 946
    /4 switch, 123, 946
    /8 switch, 946
    /B switch, 802, 946
    /F switch, 123, 946
    /N switch, 946
    /Q switch, 947
    /S switch, 127, 946
    /T switch, 946
    /U switch, 134, 947
    /V switch, 946
    DOS 5 and 6, 134
    exit status values, 948, 1060-1061
    hard drive, 123-124
    messages, 949-952
    safe format, 134
Format, low-level, 684-685
Format, operation of, 653-654
    safe, 543
Formatting, 117
Fragmentation, 133
Fragmented file, 135, 815, 826-827, 830
    causes of, 817-819
    correcting prior to DOS 6, 820-822, 827
    correcting with DOS 6, 822-823, 827
    detecting with CHKDSK, 819-820, 827
    illustration of, 816
    system performance and, 816
    UNDELETE command and, 532

G

General failure error reading (or writing) drive *n*: message, 1092
GOTO command, 303-306, 313, 952-953
    messages, 953
GRAFTABL command, 784-785, 796, 953-955
    /? switch, 954
    /STATUS switch, 954
    code-page values, 954
    exit status values, 954, 1061
    messages, 955
GRAPHICS command, 234-236, 237, 955-957
    /B switch, 956
    /LCD switch, 956
    /PRINTBOX switch, 956
    /R switch, 956

- messages, 957
- printer types, 955-956
- printers supported by, table of, 235

GREENFOR.BAT batch file to select a green foreground color, 664

GREET.BAT batch file that displays a greeting based on the current time, 286-287, 559

H

Hardcopy, 229

Hardware, 1, 30
- illustration of common parts, 2
- table of common components and their functions, 3

Head crash, 680

HELLO.BAT batch file to display a personalized greeting, 270-271

HELLO.COM program created with DEBUG to print "Hello, world!" on screen, 554-556

HELLO.SCR debug script file to create HELLO.COM, 558

Help
- contents of screen, table of, 57
- DOS 5 online, 54-55, 780-782
- DOS 6 online, 55-57, 782-784
- example of, 56
- main menu, illustration of, 56

HELP command, 781-782, 797, 957-958
- messages, 958

HELPDOS.BAT batch file that displays help for key DOS commands, 305-306

Hidden files, 40, 440, 441
- displaying with ATTRIB command, 440

High-level format, 688

HIMEM.SYS device driver, 621, 640-642
- /A20CONTROL switch, 641
- /CPUCLOCK switch, 642
- /HMAMIN switch, 640
- /INT15 switch, 640
- /MACHINE switch, 641
- /NUMHANDLES switch, 640
- /SHADOWRAM switch, 640

I

I/O redirection, 365

IBMBIO.COM file, 40

IBMDOS.COM file, 40

IDE (Integrated Drive Electronics) disk controller, 27

IF command, 280-287, 291-292, 959-960
- comparing character strings, 287
- IF ERRORLEVEL, 283-287, 291
  - testing for a specific value, 286
- IF EXIST, 279-283, 291
- messages, 960
- NOT operator, 281-283

IMPTROOT.BAT batch file to display the contents of important files, 281-282

Incorrect DOS version message, 1092

Incorrect number of parameters message, 1093

Incremental backup, 416

Input redirection operator (<), 352-353, 357

INS key, 88

Insert disk with \COMMAND.COM in drive *n*: and strike any key when ready message, 1093

Insert disk with batch file and press any key when ready message, 1093

Insert mode, 88

Insufficient disk space message, 1093

Insufficient memory message, 1093-1094

Interactive processing, 259, 277

Interleaving, disk sector, 682-684
- illustration of, 684

INTERLNK command, 692-694, 961-963
- /AUTO switch, 692, 961

    - /BAUD switch, 692, 961
    - /COM switch, 692, 962
    - /DRIVES switch, 692, 961
    - /LOW switch, 692, 962
    - /LPT switch, 692, 962
    - /NOPRINTER switch, 692, 961
    - /NOSCAN switch, 692, 962
    - /V switch, 692, 962
    - controlling drive redirection with, 693-694
    - copying files to another system, 695-697
    - establishing a connection with, 692-693
- INTERLNK device driver, 692-693, 696
    - device names, illustration of, 691
- Intermediate file error during pipe message, 1094
- Internal commands, 51, 61
    - table of, 54
- Internal drive, 115
- Internal stack overflow, system halted message, 1094
- International configuration, setting, 716-722
- Interrupts providing support for hardware, 457-458
- INTERSVR command, 694, 696, 963-965
    - /B switch, 964
    - /BAUD switch, 964
    - /COM switch, 964
    - /LPT switch, 964
    - /RCOPY switch, 964
    - /V switch, 964
    - /X switch, 963
    - copying files to another system, 695-697
    - creating a server with, 694
    - messages, 965
- Invalid argument message, 1094
- Invalid COMMAND.COM in drive *n*: message, 1094
- Invalid drive in search path message, 1094
- Invalid drive or file name message, 1095
- Invalid drive specification message, 1095
- Invalid number of parameters message, 1093, 1095
- Invalid parameter combination message, 1095
- Invalid parameter message, 1094, 1095
- Invalid path message, 1095
- Invalid switch message, 1095
- Invalid syntax message, 1096
- IO.SYS file, 40-41
- IRQ settings, displaying current with MSD command, 812

J

- JOIN command, 496-497, 498, 965-966
    - /D switch, 965
    - commands not to use with, table of, 493
    - messages, 966
- Joystick, displaying settings with MSD command, 810
- Jumper, 240
    - illustration of, 241
- JUNE.BAT batch file to display a June 1993 calendar, 273

K

- Kb (Kilobyte), 13, 30
- Key combination, 103
- KEYB command, 720-722, 727-728, 966-970
    - /E switch, 720, 967
    - /ID switch, 720, 967
    - exit status values, 1061, 968
    - keyboard codes, table of, 967-968
    - messages, 968-970
- Keyboard, 1
    - connecting, illustration of, 4
    - copying files from, 90

- cursor-control keys, 91-92
  - illustration of, 92
- function keys, 86-89
- key combinations, 92, 94, 101
- layouts, 1069-1075
  - illustration of common, 86
- making enhanced behave as conventional, 459-460
- primary sections, 85
  - illustration of layout, 87
- typewriter keys, 86

Keyboard scancode values, table of, 666-668
Keyboard template, 720
- commands prior to DOS 3.3, table of, 722
- international, table of, 721
- selecting, 720-722

Keypad, numeric, 91
Keys, redefining with ANSI.SYS, 665-669
Kilobyte (Kb), 13, 30

L

Label, 320
LABEL command, 585-586, 593, 970-973
- messages, 972-973

Landing zone, 681
- illustration of, 681

List device, 257
LISTPAR.BAT batch file that displays the value of its arguments, 293
LISTTREE.BAT batch file that saves and restores the current directory, 320
LOADFIX command, 630-631, 636, 973
LOADHIGH command, 635, 974-975
- messages, 975

Logical disk drive, 491, 503, 844
Lost cluster, 481, 489
Low-level format, 684-685, 686, 688
- function of, 685

LPT (line printer port), 231

M

Macro (DOSKEY), 520
Master boot record, 39
- illustration of, 844

Megabyte (Mb), 14, 30
MEM command, 623-627, 635, 975-977
- /CLASSIFY switch, 629-630, 975
- /DEBUG switch, 624-626, 975
- /FREE switch, 626-627, 975
- /MODULE switch, 627, 975
- /PAGE switch, 975
- /PROGRAM command, 975
- messages, 977

MEMMAKER command, 631-633, 977-979
- /BATCH switch, 631, 978
- /SWAP switch, 631, 978
- /UNDO switch, 631, 978
- /W switch, 631, 978
- CONFIG.SYS and, 462

Memory, expanded (*See also* Expanded memory), 615-619
Memory, extended (*See also* Extended memory), 619-621
Memory-resident commands, 252
- installing, 458-459

Memory-resident software, 257
Memory usage, viewing with MSD command, 808
MENU.BAT batch file that displays a menu, 317-319, 561-562, 564-565
MENU.BAT batch file menu system using escape sequences, 674-675
Metacharacter, 775, 814
Microsoft Windows, invoking in AUTOEXEC.BAT, 326
MIRROR command, 530, 532, 534, 979-982
- /1 switch, 979
- /PARTN switch, 549-550, 979
- /T switch, 979
- /U switch, 534, 979
- messages, 980-982

    - saving information for UNFORMAT with, 536
    - saving master boot record with, 549-550
- MKDIR (MD) command, 176-180, 183, 982-983
  - messages, 983
- MODE command, 786-794, 984-991
  - code page
    - commands, summary of, 728
    - displaying status information with, 725
    - preparing with, 723-724
    - refreshing with, 725-726
    - selecting with, 724-725
  - device status, displaying with, 794
  - keyboard
    - improving responsiveness with, 607
    - setting response rate with, 788
  - messages, 988-991
  - printer output, customizing with, 786-787
  - printer timeout errors, retry options for, table of, 787
  - serial port, configuring with, 791-794
  - typematic delay time values, table of, 789
  - typematic repeat rate values, table of, 788
  - video display, customizing with, 789-791
    - settings supported by, 790
- Modem, 241
- Monitor, function of, 2
- MORE command, 991-992
  - pipe operator and, 369-370
  - and redirection and, 353
- Motherboard, 24
  - illustration of, 29
- Mouse, 2
- Mouse settings, displaying with MSD command, 810
- MOVE command, 190-191, 992-993
  - message, 993
- MSAV command, 994-996
  - /A switch, 994
  - /C switch, 994
  - /F switch, 994
  - /IM switch, 995
  - /L switch, 994
  - /LE switch, 995
  - /N switch, 994
  - /NGM switch, 995
  - /P switch, 994
  - /PS2 switch, 995
  - /R switch, 994
  - /S switch, 994
  - /VideoSetting switch, 994-995
- MSBACKUP command, 396-406, 996-997
  - /BW switch, 996
  - /LCD switch, 996
  - /MDA switch, 996
  - /T switch, 996
  - backup options, 414-415
  - backup sets, 402-404
  - catalog file extensions, table of, 402
  - catalog filename character meanings, table of, 403
  - catalogs, 402-404
  - compare operations, 415-416
  - configuring, 396-397
  - messages, 997
  - performing a daily backup with, 401-402
  - performing a monthly backup with, 400-401
  - restore options, 415
  - restoring files with, 404-406
    - complete disk, 405-406
    - specific files or directories, 406

- selecting and excluding specific files with, 403-404
- setup file, 398
  - creating, 398-400

MSD command, 806-813, 997-998
- /B switch, 998
- /F switch, 998
- /I switch, 998
- /P switch, 998
- /S switch, 998
- COM ports option, 812
- Computer option, 807
- Device drivers option, 813
- Disk drives option, 811
- IRQ status option, 812
- LPT ports option, 811
- main menu, 806-807
- Memory option, 808
- Mouse option, 810
- Network option, 809
- OS version option, 809
- Other adapters option, 810
- TSR programs option, 813
- Video option, 808

MSDOS.SYS file, 40-41

MULTIDEL.BAT batch file that lets you delete one or more files, 316, 317

N

Named parameter, 294-296, 301

Nested batch file, 311, 314, 320

NESTED.BAT batch file example of nesting files, 311-312

Network information, displaying with MSD command, 809

NLSFUNC command, 729, 999
- messages, 999

No such file or directory message, 1096

NOEND.BAT batch file example of nesting files, 311-312

NOMSG.BAT batch file example of copying files with output redirection, 362

Non-system disk, 42

Not enough memory message, 1093-1094, 1096

Not ready error reading (or writing) drive *n*: message, 1096

NUL device, redirecting output to, 361-362

Null string, 287
- representing in batch files, 290

NULL-modem cable, 698

Numeric keypad, 91
- illustration of, 91

O

Operating system, 18, 30
- displaying specifics about with MSD command, 809

Out of environment space message, 1097

Output, redirecting to NUL device, 361-362

Output redirection operator (>), 347-350, 357

Overstrike mode, 88

P

Parallel communication, 814

Parallel device, 241
- names, table of, 232

Parallel ports, displaying information about with MSD command, 811

Parameter value not allowed message, 1096

Parameter value not in allowed range message, 1096

Parameters not compatible message, 1095, 1096

Parent directory, 215

PARKC.SCR debug script file, 682

Parking, disk head, 679-682

Partition, 832, 844
- extended, 832, 844
- illustration of, 832

  - illustration of primary and extended, 833
  - primary, 832, 844
  - reasons to use, 833
- PARTNSAV.FIL file containing saved master boot record, 550-551
- PATH command, 198-201, 207, 1000-1001
  - AUTOEXEC.BAT and, 200-201, 322-323
  - messages, 1001
- Path not found message, 1095, 1096
- Path too long message, 1097
- PAUSE command, 266-268, 269-270, 1001-1002
  - messages, 1002
- Performance, 612
  - BREAK command and, 607
  - directory considerations for maximum, 599-601
  - disk considerations for maximum, 602-603
  - file considerations for maximum, 601-602
  - improving keyboard responsiveness with MODE command, 607
- Physical drive parameters, changing, 460
- Pipe operator (|), 367-373, 381
  - batch file restrictions using, 380
  - combining with redirection operators, 371-372
  - common problems using, 372
  - locating on keyboard, 367-368
  - TEMP environment entry and, 379
  - temporary files used by, 378-379
- POWER command, 795, 1003
  - messages, 1003
- POWER.EXE device driver, 795
- Power-on self-test, 33
- Power supply, 24-25
  - illustration of typical, 25
- Pretender command, 503
- Previous-command buffer, 520
- Primary DOS partition, 42, 832, 844
- PRINT command, 244-256, 1004-1007
  - /B switch, 254, 1004
  - /C switch, 242, 247, 1004
  - /D switch, 255, 1004
  - /M switch, 255, 1004
  - /P switch, 1004
  - /Q switch, 245, 253, 1004
  - /S switch, 255, 1004
  - /T switch, 246-247, 1004
  - /U switch, 256, 1005
  - in AUTOEXEC.BAT, 324
  - canceling, 246
  - CPU time and, 255-256
  - memory allocated by, 257
  - messages, 1006-1007
  - multiple files and, 245-246
- Print queue, 257
- Printer
  - copying files to, 232-233, 243
  - function of, 2
  - identifying type, 230
  - parallel, 230
  - serial, 230
- Printer plugs, illustration of, 5
- PRINTER.SYS device driver, 718, 727
  - printer types supported by, table of, 718
- Program list, 762
- Program too big to fit in memory message, 1097
- PROMPT command, 173, 775-778, 796, 1008-1009
  - controlling screen colors with, 664-665
  - in AUTOEXEC.BAT, 323
  - metacharacter translations, table of, 776
  - metastrings, 1008
- PRTSC.COM program, 568

PS2PARKC.SCR debug script file, 681

Q

Question mark wildcard (?), 161-163, 164
Queue, 245

R

RAM drive, creating, 621-623
RAMDRIVE.SYS device driver, 621-622, 634
  /A switch, 622
  /E switch, 622
  TEMP environment entry and, 622
Read-ahead, 448, 472
Read-only file, 420, 441
Real-time compression, 855
RECOVER command, 591-593, 594, 1009-1011
  messages, 1010-1011
REDBACK.BAT batch file to select a red screen background, 663
Redirection operators, 347
  combining input and output, 356
  combining with pipe operator, 371-372
  problems when using, 351-352
REM command, 266, 269, 1011-1012
  in CONFIG.SYS, 451-453
RENAME (REN) command, 143-144, 146, 1013-1014
  messages, 1014
REPLACE command, 431-433, 435, 1014-1017
  /A switch, 1014
  /P switch, 433, 1014
  /R switch, 1014
  /S switch, 432, 1014
  /U switch, 1015
  /W switch, 1015
  command-line switches, table of, 433
  exit status values, 1061-1062
  messages, 1016-1017
Reserved memory, 627, 645
Resident command processor, 346
RESTORE command, 391-394, 1018-1022
  /A switch, 411, 1018
  /B switch, 411, 1018
  /D switch, 412, 1018
  /E switch, 411, 1018
  /L switch, 411, 1018
  /M switch, 412, 1018
  /N switch, 412, 1018
  /P switch, 411, 1018
  /S switch, 392, 1018
  exit status values, 414, 1062, 1019
  in batch files, 413-414
  large files and, 412
  messages, 1020-1022
  restoring an entire hard disk, 391-392
  restoring a specific file or directory, 392-394
RMDIR (RD) command, 181-182, 183, 1022-1023
  message, 1023
Root directory, 171-172, 182, 191
  calculating number of entries, 130, 652-653
  location of on disk, 651-653
  number of sectors reserved for, table of, 130, 652
  reserving space for, 129-130
Rotational delay, 829, 830

S

Safe format, 543
SAVEDIR.BAT batch file that saves the current directory, 319-18
SCANCODE.SCR debug script file, 560-561
Screen (*See also* monitor), 2
Screen colors

- controlling with ANSI.SYS, 663-665
- controlling with the PROMPT command, 664-665

Screen editor, 714
Screen printing, 233-235
- graphics, 234-235
- with serial printer, 233-234

Scroll bars, 705, 714
SCSI (Small Computer System Interface) disk controller, 27
Sector, 118, 135
- number per track for common disk sizes, table of, 120

Sector interleave, 688
Sector not found reading (or writing) drive *n*: message, 1097
Seek error reading (or writing) drive *n*: message, 1097
Seek time, 829, 830
SELERASE.BAT batch file providing selective delete capabilities, 362-365
Serial communication, 814
Serial device, 241
- configuring with the MODE command, 791-794
- names, table of, 232

Serial ports
- communications settings, 231-232
- displaying information about with MSD command, 812

Server, 689, 696, 698
SET command, 333-336, 339, 1023-1025
- changing path and prompt with, 345
- in CONFIG.SYS, 472
- messages, 1025

SETVER command, 804-806, 1025-1028
- /DELETE switch, 1026
- /QUIET switch, 1028
- exit status values, 1026, 1062-1063
- messages, 1027-1028

SETVER.EXE device driver, 804
SHARE command, 589-590, 594, 1028-1029
- /F switch, 590, 1028
- /L switch, 590, 1028
- in AUTOEXEC.BAT, 326
- large disk partitions and, 590
- messages, 1029

SHIFT command, 308-311, 314, 1029-1030
SHIFTALL.BAT batch file that displays all of its named parameters, 310
SHIFTONE.BAT batch file that uses the SHIFT command, 309
SHIFTTWO.BAT batch file that uses the SHIFT command twice, 309-310
SHIFT-PRTSC (print screen), 233, 237
SHOWALL.BAT batch file that displays the value of its arguments, 288
SHOWENV.BAT batch file that uses named parameters to display the current command path and system prompt, 295-296
SHOWLINE.COM program, 562, 563-564
SKIPDIR.BAT batch file that uses GOTO, 304
SMARTDRV command, 605-606, 1030-1033
- /B switch, 1031
- /C switch, 605, 1031
- /E switch, 1031
- /L switch, 1031
- /Q switch, 1031
- /R switch, 1031
- /S switch, 605-606, 1031
- in AUTOEXEC.BAT, 326
- buffer sizes, 1032
- messages, 1033

SMARTDRV.SYS device driver, 604, 634
- /A switch, 604
- InitialCacheSize parameter, 604
- MinimumCacheSize parameter, 604

    reducing number of disk buffers when using, 606
Software, 17, 19, 30
SORT command, 1033-1035
    /+n switch, 1033
    /R switch, 1033
    command line switches, table of, 355
    pipe operator and, 368-369
    redirection and, 354-355
SORTIT.BAT batch file example of using pipe operator, 380
Source disk, 227
Source file, 152
Specified COMMAND search directory bad message, 1097
ST (Shugart Technology) disk controller, 27
Stack, 472
Standard times and 24-hour equivalents, table of, 47
stdaux (auxiliary communications device), 360, 365
stderr (error message destination), 360, 365
stdin (standard input device), 360, 365
stdout (standard output device), 360, 365
stdprn (printer), 360, 365
Storage space, calculating amount on a disk, 119-120
SUBST command, 493-495, 498, 1035-1037
    /D switch, 1035
    abbreviating long directory names with, 495
    commands not to use with, 493, 1036
    creating logical disk drives with, 502
    messages, 1037
Surge suppressor, 4-6
    illustration of common types, 6
Switch, 70
Symbol set, selecting, 718-720
SYS command, 801-803, 1037-1039
    messages, 1038-1039
SYSINIT program, 41
System, displaying specifics about with MSD command, 807
System date
    displaying current, 49-50
    setting, 49-50
System disk, 42
    building a bootable, 127-128
System file, 441
System files, transferring to a target disk, 801-803
System prompt, customizing, 775-778
System startup, controlling, 471
    F5 key pressed during, 471
    F8 key pressed during, 471
System time
    correcting an invalid entry, 48
    displaying current, 46-47
    setting, 47-49
    specifying a precise value, 48-49

T

Target disk, 227
Task swapper, 762
TEMP environment entry, defining in AUTOEXEC.BAT, 324-325
Terminate batch job (Y/N)? message, 1097-1098
Terminate-and-stay-resident (TSR) commands, 252
TESTAUTO.BAT batch file that tests if AUTOEXEC.BAT is in the root directory, 292
TESTFILE.BAT batch file that uses a batch parameter to test if a given file is in the current directory, 292-293
Text file, 152
TIME command, 46-49, 58, 1039-1040
    messages, 1040

Time stamp, 68, 74, 83
Too many parameters message, 1093, 1098
Too many open files message, 1098
Tracks, 118, 135
  number per side for common disk sizes, table of, 120
Transient command processor, 346
TREE command, 193-197, 207, 1040-1041
  /A switch, 1041
  /F switch, 195-196, 1041
  messages, 1041
TREEPRT.BAT batch file to invoke the TREE command, 271-272
TSR programs, displaying information about with MSD command, 813
TYPE command, 137-139, 146, 1042-1043
TYPE3.BAT batch file that displays the contents of three files, 307
TYPEALL.BAT batch file that displays the contents of multiple files, 310-311
Typematic delay, 814
Typematic rate, 788
Typing error, correcting, 45-46

U

UNDELETE command, 521-535, 544, 1043-1046
  /ALL switch, 528-529, 1043
  /DOS switch, 1043
  /DT switch, 1043
  /LIST switch, 527-528, 534-535, 611-612, 1043
  /LOAD switch, 1043
  /PURGE switch, 1044
  /S switch, 1044
  /T switch, 532, 1044
  data sentry protection, 535-536
  file tracking with, 530-534, 544
  maximum number and size of files, table of, 533
  fragmented files and, 532
  messages, 1045-1046
Undelete, 552
  listing files available to, 527-528
  process, 526
Unformat, 536-543, 552
UNFORMAT command, 536-543, 544-545, 1046-1048
  /J switch, 537, 538, 1047
  /L switch, 1047
  /P switch, 542, 1047
  /PARTN switch, 550-552, 1047
  /TEST switch, 540, 542, 1047
  /U switch, 542-543
  messages, 1048
  with MIRROR support, 537-540
  without MIRROR support, 540-543
Uninstall disk, and DOS 6 upgrade, 1087
Upgrading to DOS 6, 1083-1087
Upper memory blocks (UMB), 627-629

V

VER command, 44-45, 57, 1048-1049
VERIFY command, 590-591, 594, 1049-1051
  messages, 1051
Version number, reporting a specific with DOS 5 or later, 803-806
Video configuration, displaying with MSD command, 808
Video plugs, illustration of, 5
VOL command, 586-587, 593, 1051-1052
  messages, 1052
Volume label, 122, 135, 598
VSAFE command, 1052-1053
  /A switch, 1053
  /C switch, 1053
  /D switch, 1053

/N switch, 1053
/NE switch, 1053
/NX command, 1053
/U switch, 1053
in AUTOEXEC.BAT, 326

W

Warm boot, 93, 101, 103
WHITEBLU.BAT batch file to select a white background and blue foreground, 664
WIDEDIR.BAT batch file to display wide format directory listing, 270
WILD.BAT batch file that adds wildcard support to commands that don't support wildcards, 167-168
Wildcard characters, 149-151, 153, 164, 169
Windows, illustration of, 35
Write protect error writing drive *n*: message, 1098
WRONGIF.BAT example of incorrect use of IF ERRORLEVEL command, 285

X

XCOPY command, 226-227, 421-424, 434, 1054
/A switch, 1054
/D switch, 1054
/E switch, 1054
/M switch, 422-423, 1054
/P switch, 1054
/S switch, 423, 1054
/V switch, 1054
/W switch, 1054
command line switches, table of, 424
exit status values, 1063
messages, 1056-1057

Y

Y_OR_N.COM program, 559-354
Y_OR_N.COM gets a yes or no response, 296-297
YELLBLUE.BAT batch file to select a yellow screen background and a blue foreground, 672

IBM PC & Compatibles/Operating Systems

Skill Level Guide

☐ Beginner
☐ Intermediate
☐ Advanced
☒ For Every User

- *46 chapters packed with answers to every conceivable DOS question.*
- *Complete Command Reference helps you pinpoint information quickly.*
- *Provides up-to-the minute information on all the enhancements of DOS 6.*

DOS

the
COMPLETE REFERENCE

FOURTH EDITION

The Best-Selling One-Volume Resource You Can Count On to Answer Every Question About DOS

If you're a new user of DOS, or are seeking the best information on the new features of DOS 6, this exhaustive volume is the ONLY DOS book you'll need on your shelf.

Noted PC expert Jamsa provides all the information you need to start using DOS productively, beginning with the basics, in a clear, easily-understood manner. If you have some DOS experience, you'll find secrets and tips available in no other book, along with more advanced behind-the-scenes concepts.

- **Get moving quickly by following the Hands On sessions in the introductory chapters.**
- **Find out more than you ever imagined about how your keyboard, disk drives, and diskettes work with DOS.**
- **Discover how to get the most from DOS directories, and from printers and other peripherals.**
- **Learn how to best protect your files, avoid crises, debug and recover disk errors, and how to prevent viruses.**
- **Move on to advanced topics, such as DOS memory management**
- **Find the answers to specific questions in no time by referring to the Command Reference guide.**

Loaded with illustrations, notes, and exercises, **DOS: The Complete Reference, Fourth Edition,** guarantees you'll move from chapter to chapter as smoothly as possible. No matter what version of DOS you use, this comprehensive reference assures you of getting the most from DOS — whether you are exploring its fundamentals or have advanced to its most sophisticated capabilities.

ISBN 0-07-881904-0
90000
9 780078 819049

$34.95

Best-Seller! Over 350,000 Copies Sold

Covers DOS Versions 1.0, 2.0, 2.01, 2.1, 3.0, 3.1, 3.2, 3.3, 4.0, 4.01, 5.0 and 6.0